AJAX Database Application

Agenda

1 **Ajax Database Application**

Objectives

At the end of this module, you will be able to:

- Learn how to communicate with database using AJAX and DAO
- Learn how to return a JSP page as AJAX response

Ajax Database Application

AJAX Database Application

Case Study :

Create a web application that takes new entry for department and check if the department is already entered in the database

- The entry form should be designed in a **JSP page**
- Department no should be **autogenerated**
- While taking the input in department name check using ajax if the name is already present in the database using **DAO**
- Notify the message in a span placed adjascent to the department name textbox

Step 1- DeptEntry.jsp

```
<%@page import="com.wipro.dao.DeptDAO"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
  charset=UTF-8">
        <title>Department Entry</title>
    </head>
    <body>
        <script src="DeptApps.js"></script>
        <h1>Enter Department Details</h1>
        <%
            DeptDAO dao=new DeptDAO();
        %>
```

Step 1- DeptEntry.jsp (Contd.).

```
<form method="post" action="AddDepartmentServlet">
            <table>
                <tr>
                 <td>Deptno</td>
                 <td><input type="text" name="dno"
 value="<%=dao.generateDeptno()%>" readonly/></td>
                </tr>
                <tr>
                 <td>Enter Deptname</td>
                 <td><input type="text" name="dname"
 onblur="sendRequest(this.value)"/></td>
                 <td><span id="dname_status"></span></td>
                </tr>
<tr><td>Enter Location</td>
    <td><input type="text" name="loc"/></td>
</tr>
```

Step 1- DeptEntry.jsp (Contd.).

```
 <tr>
 <td><input type="submit" name="b1" value="Add"/></td>
 <td><input type="reset" name="b2" value="Clear"/></td>
 </tr>
 </table>
 </form>
 </body>
</html>
```

Step 2- DeptApps.js (javascript)

```
var req;//global variable
//function to get the department name as parameter
//and passing it to server for checking its existence
function sendRequest(dnm)
{
//for firefox/safari/opera/google chrome
if (window.XMLHttpRequest) {
req = new XMLHttpRequest( );
}
else if (window.ActiveXObject) //for IE
{
req = new ActiveXObject("Microsoft.XMLHTTP");
}
//concatenate the dname as parameter value to url
var url = "DnameChecker?dname="+dnm;
```

Step 2- DeptApps.js (javascript) (Contd.).

```
req.onreadystatechange = getResponse;//check server request
  state
req.open("POST", url, true);//send request to server
req.send(null);
}
//function to get the response and display in the specific
  area
function getResponse()
{
if (req.readyState==4) //request is complete
{
if (req.status == 200) //target page is found
{
//write the response text in the span area
document.getElementById("dname_status").innerHTML =
  req.responseText;;
}}}
```

Step 3- DeptDAO.java (DAO)

```
package com.wipro.dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *  Dept DAO class
 */
public class DeptDAO
{
    //Database connection url
    String url="jdbc:oracle:thin:@localhost:1521:orcl";
```

Step 3- DeptDAO.java (DAO) (Contd.).

```
//Database connection username
    String username="scott";
    //Database connection password
    String password="tiger";

//method to establish database connection
    public Connection connect() throws SQLException,
  ClassNotFoundException
    {
        //Using type 4 driver
        Class.forName("oracle.jdbc.OracleDriver");
        //return connection url
 return DriverManager.getConnection(url,username,password);
    }
```

Step 3- DeptDAO.java (DAO) (Contd.).

```
//method to autogenerate deptno by finding the maximum
  deptno
     //currently present and generating the new deptno by 10
     public int generateDeptno()
     {
          int deptno=0;
          try
          {
 //establish conection
    Connection conn=connect();
 //query to fetch the max deptno value
     String query="select max(deptno) from dept";
 //Using PreparedStatement the query plan is created
     PreparedStatement ps=conn.prepareStatement(query);
//fetch value into resultset
     ResultSet rs=ps.executeQuery();
```

Step 3- DeptDAO.java (DAO) (Contd.).

```
if(rs.next())
                {
                    //fetch max(deptno) value into deptno
                    deptno=rs.getInt(1);
                }
                //increment deptno by 10
                //if no value is returned from the query ie
          //when table is empty then deptno will start with 10
                deptno=deptno+10;
           }
           catch(Exception ex)
           {
                ex.printStackTrace();
           }
           return deptno;
      }
```

Step 3- DeptDAO.java (DAO) (Contd.).

```
//method to check if dname passed as parameter is already
  present
    public boolean getDname(String dname)
    {
        try
        {
            //establish conection
            Connection conn=connect();
            //query to fetch the max deptno value
          String query="select * from dept where dname=?";
       //Using PreparedStatement the query plan is created
        PreparedStatement ps=conn.prepareStatement(query);
//send dname as parameter to preparedstatement
ps.setString(1, dname);
```

Step 3- DeptDAO.java (DAO) (Contd.).

```
//fetch value into resultset
            ResultSet rs=ps.executeQuery();
            //check if resultset contains at least one row
            if(rs.next())
            {//dname is present
                return true;}
            else
            {//dname is not present
                return false;}
        }
        catch(Exception ex)
        {//for exception also return false
            return false;}
    }
}
```

Step 4- DeptChecker.java (Servlet)

```
package com.wipro.servlet;

import com.wipro.dao.DeptDAO;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Servlet class which receives the
 * application request
 */
```

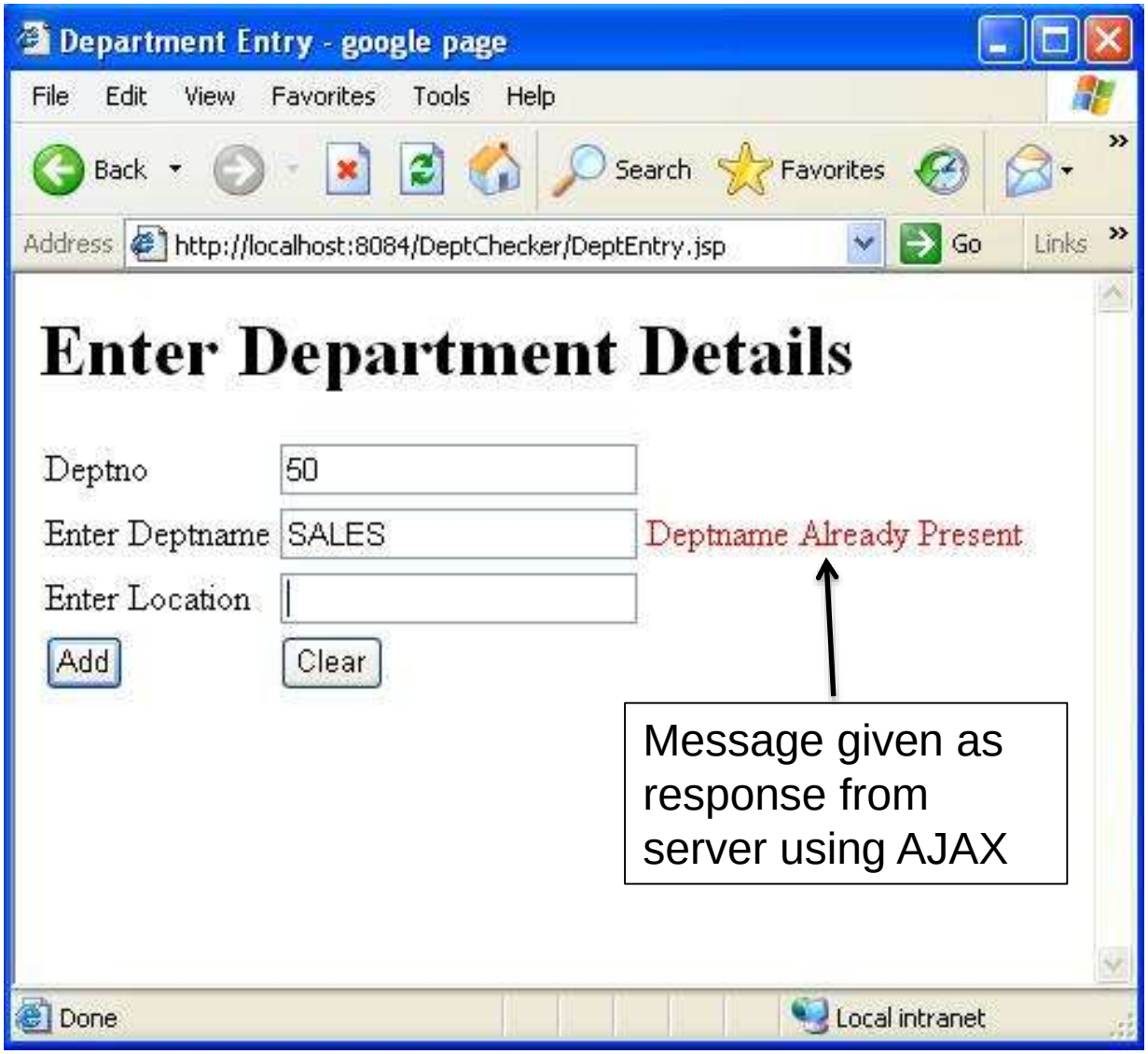
Step 4- DeptChecker.java (Servlet) (Contd.).

```
public class DnameChecker extends HttpServlet {
      protected void doPost(HttpServletRequest request,
  HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            //extract department name from the parameter
            String deptname=request.getParameter("dname");
            if(deptname.length()>0)//if not blank
            {
                DeptDAO dao=new DeptDAO();
                //check from DAO if department exists
  if(dao.getDname(deptname)==true)//when found
 {out.println("<font color=red>Deptname Already
  Present</font>");
                }
```

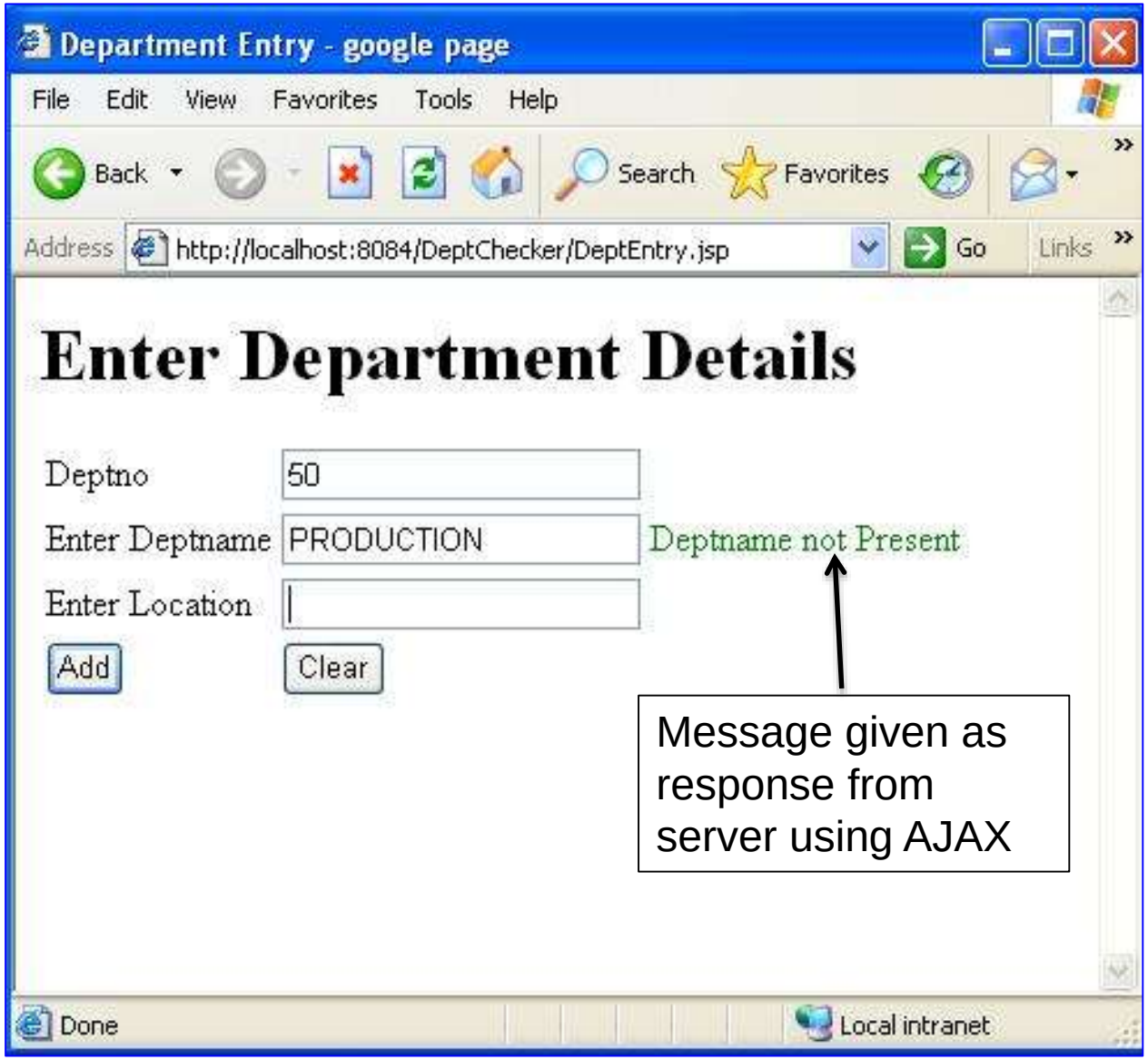
Step 4- DeptChecker.java (Servlet) (Contd.).

```
else
                {//when not found
                    out.println("<font color=green>Deptname
  not Present</font>");
                }
            }
            else
            {//for blank
                out.println("<font color=red>Deptname
  cannot be blank</font>");
            }
        } finally {
            out.close();
        }
    }
}
```

Expected Output (One) – preexisting deptname



Expected Output (Two) – new deptname



AJAX Database Application returning JSP

Case Study :

Create a web application that provides the user a list of jobs, selecting which the employees having that job profile is displayed

- When a particular job is chosen, use ajax to fetch the list of employees having that job profile and display them in a div area
- Fetch the employees from the database using **DAO** and store them as a Vector of **EmployeeBean** objects
- Store the Vector in a **session**
- The final response is generated from a **JSP** which retieves the employee records from the session and dispay them in a tabular manner

Step 1- EmployeesJob.html (HTML)

```
<html>
    <head>
        <title>View Employees</title>
    </head>
    <script src="EmpApps.js">
    </script>
    <body>
        Choose Employee job type
<select name="job_list" onchange="sendRequest(this.value)">
            <option value="nojob">---Choose Job---</option>
            <option value="ANALYST">Analyst</option>
            <option value="CLERK">Clerk</option>
            <option value="SALESMAN">Salesman</option>
            <option value="MANAGER">Manager</option>
        </select>
<hr/><div id="emp_details"></div>
    </body></html>
```

Step 2- EmpApps.js (Javascript)

```
var req;//global variable
//function to get the job as parameter and
//passing to server to find related employee records
function sendRequest(j)
{
//for firefox/safari/opera/google chrome
if (window.XMLHttpRequest) {
req = new XMLHttpRequest( );
}
else if (window.ActiveXObject) //for IE
{
req = new ActiveXObject("Microsoft.XMLHTTP");
}
//concatenate the job as parameter value to url
var url = "FindEmp?job_list="+j;
```

Step 2- EmpApps.js (Javascript) (Contd.).

```
req.onreadystatechange = getResponse;//check server request
  state
req.open("POST", url, true);//send request to server
req.send(null);
}
//function to get the response from jsp
//and display in the specific area
function getResponse()
{
if (req.readyState==4) //request is complete
{
if (req.status == 200) //target page is found
{
//write the response text in the div area
document.getElementById("emp_details").innerHTML =
  req.responseText;;
}}}
```

Step 3- EmployeeBean.java (Bean)

```
package com.wipro.bean;

import java.io.Serializable;
/**
 * Employee Bean class
 * containing accessor and mutator
 * methods of Employee attributes in the table
 */
public class EmployeeBean implements Serializable
{
    private int empno;
    private String ename;
    private String job;
    private double salary;
    public EmployeeBean()
    {
    }
```

Step 3- EmployeeBean.java (Bean) (Contd.).

```
    public int getEmpno() {
        return empno;          }
    public void setEmpno(int empno) {
        this.empno = empno;    }
    public String getEname() {
        return ename;          }
    public void setEname(String ename) {
        this.ename = ename;    }
    public String getJob() {
        return job;     }
    public void setJob(String job) {
        this.job = job; }
     public double getSalary() {
        return salary;         }
    public void setSalary(double salary) {
        this.salary = salary; }
}
```

Step 4- EmployeeDAO.java (DAO)

```
package com.wipro.dao;

import com.wipro.bean.EmployeeBean;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.Vector;

/**
 * DAO class to fetch Employee records
 * on the basis of selected job profile
 */

public class EmployeeDAO
{
```

Step 4- EmployeeDAO.java (DAO) (Contd.).

```
//Method to fetch employee records who are having the job
//matching the job supplied from the view as parameter

  public Vector<EmployeeBean> fetchEmployees(EmployeeBean
  empbean)
     {
          //Database connection url
          String url="jdbc:oracle:thin:@localhost:1521:orcl";
          //Database connection username
          String username="scott";
          //Database connection password
          String password="tiger";
          //Vector to store a collection of employee objects
          //fetched from database
          Vector<EmployeeBean> V=new Vector<EmployeeBean>();
```

Step 4- EmployeeDAO.java (DAO) (Contd.).

```
try
        {
            //Using type 4 driver
            Class.forName("oracle.jdbc.OracleDriver");
  //Establish the connection
  Connection
  conn=DriverManager.getConnection(url,username,password);
   //Query to fetch empno,ename and sal based on job and
   //sort result on the basis of empno in ascending order
   String query="select empno,ename,sal from emp where job=?
   order by empno";
 //Using PreparedStatement the query plan is created
   PreparedStatement ps=conn.prepareStatement(query);
//set the job as parameter to PreparedStatement
   ps.setString(1, empbean.getJob());
```

Step 4- EmployeeDAO.java (DAO) (Contd.).

```
//Execute the query and store result in ResultSet
 ResultSet rs=ps.executeQuery();//executeQuery() for
 select
//fetch each row from resultset until no rows are
 available
            while(rs.next())
            {
  //An employee record object created to map and store
  //each attribute of employee record from the resultset
                EmployeeBean emprecord=new EmployeeBean();
emprecord.setEmpno(rs.getInt(1));//fetch empno
 emprecord.setEname(rs.getString(2));//fetch ename
 emprecord.setSalary(rs.getDouble(3));//fetch sal
 V.addElement(emprecord);//store entire employee object in
 vector
            }
```

Step 4- EmployeeDAO.java (DAO) (Contd.).

```
            }
            catch(Exception ex)
            {
                ex.printStackTrace();
            }
            return V;
        }
}
```

Step 5- FindEmp.java (servlet)

```
package com.wipro.servlet;

import com.wipro.bean.EmployeeBean;
import com.wipro.dao.EmployeeDAO;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Vector;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

/**
 * Servlet class to process application request
 */
public class FindEmp extends HttpServlet {
```

Step 5- FindEmp.java (servlet) (Contd.).

```
protected void doPost(HttpServletRequest request,
  HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {//extract job from the parameter
   String job_selected=request.getParameter("job_list");
            //check if 'nojob' is selected
            if(!job_selected.equalsIgnoreCase("nojob"))
            {
                EmployeeBean empbean=new EmployeeBean();
empbean.setJob(job_selected);//set the job
EmployeeDAO empdao=new EmployeeDAO();
                //call the fetchEmployees() of EmployeeDAO
Vector<EmployeeBean>
  employeeV=empdao.fetchEmployees(empbean);
```

Step 5- FindEmp.java (servlet) (Contd.).

```
HttpSession hs=request.getSession();
 //set the entire vector object containing employee records
  in the session
                    hs.setAttribute("emp", employeeV);
                    //redirect to ShowEmployees.jsp
                    response.sendRedirect("ShowEmployees.jsp");
                }
            }
            catch(Exception e)
            {
                out.println(e);
            }
            finally {
                out.close();
            }
        }
}
```

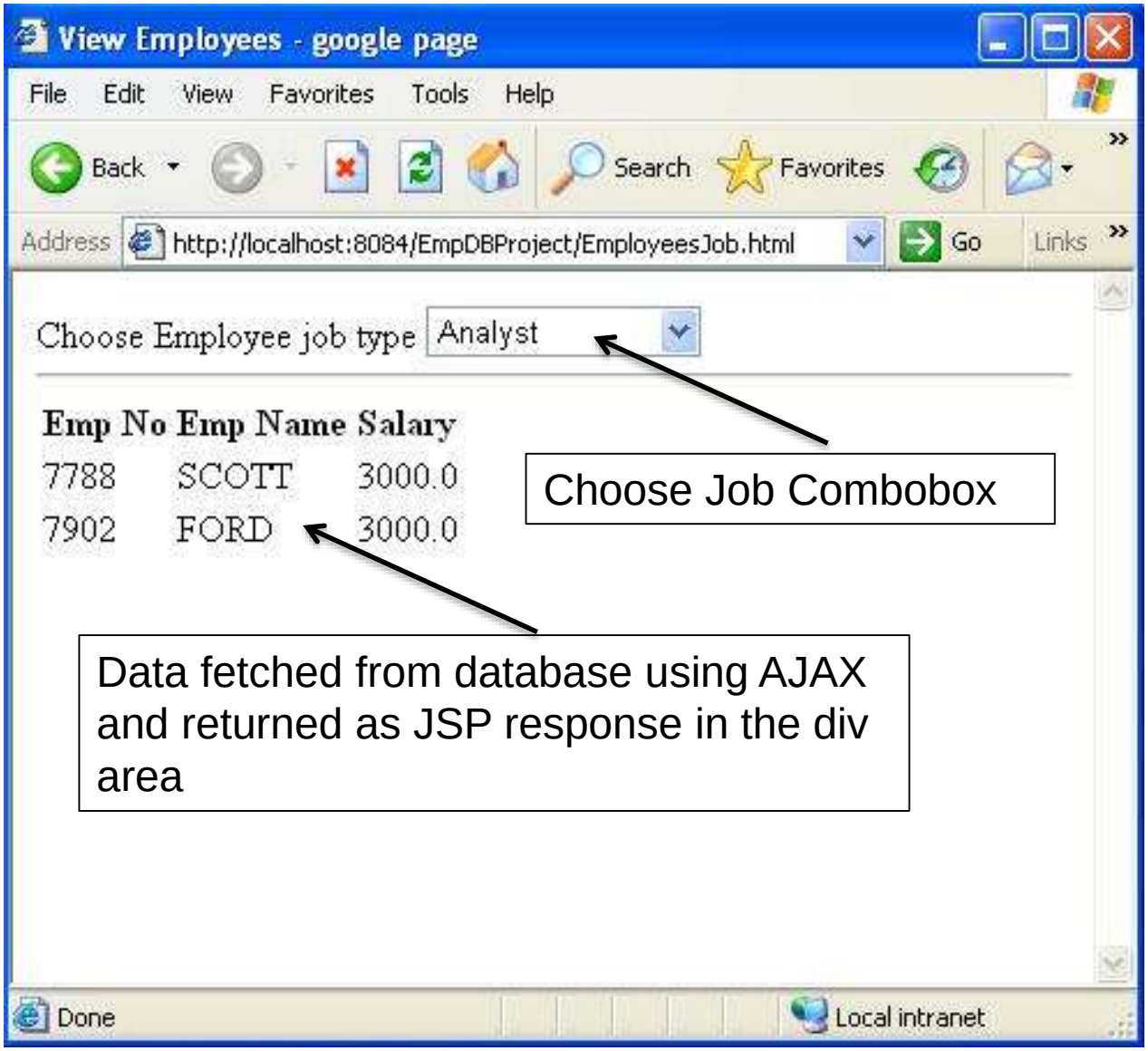
Step 6- ShowEmployees.jsp (jsp)

```
<%@page import="com.wipro.bean.EmployeeBean"%>
<%@page import="java.util.Vector"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
  charset=UTF-8">
        <title>Employees</title>
    </head>
    <body>
        <%
        //Get the Vector object containing employee records
  from session
        Vector<EmployeeBean>
  empV=(Vector<EmployeeBean>)session.getAttribute("emp");
        %>
```

Step 6- ShowEmployees.jsp (jsp) (Contd.).

```
<table>
<th>Emp No</th><th>Emp Name</th><th>Salary</th>
        <%
        //Fetch each employee bean object from vector
            for(int i=0;i<empV.size();i++){
        EmployeeBean beanobject=empV.elementAt(i);
//create rows and feed data from employee bean object
        %>
      <tr>
            <td><%=beanobject.getEmpno()%></td>
            <td><%=beanobject.getEname()%></td>
            <td><%=beanobject.getSalary()%></td>
        </tr>
        <%
            }
        %>
</table></body></html>
```

Expected Output



Summary

In this module, you were able to:

- Develop application to communicate with database using AJAX and DAO
- Develop application through which a JSP page could be returned as AJAX response

References

- w3schools.com (2012). AJAX Introduction. Retrieved April 30, 2012, from, http://www.w3schools.com/ajax/default.asp

- Greg Murray (2005). Asynchronous JavaScript Technology and XML(Ajax) With the Java Platform. Retrieved April 30, 2012, from, http://www.oracle.com/technetwork/articles/javaee/ajax-135201.html

- Adaptive path (2012). Ajax: A New Approach to Web Applications. Retrieved May 2, 2012, from, http://www.adaptivepath.com/ideas/ajax-new-approach-web-applications

Thank You