UNIVERSITY OF MAURITIUS

FACULTY OF ENGINEERING

SECOND SEMESTER EXAMINATIONS

MAY 2010

PROGRAMME	**BSc (Hons) Computer Science & Engineering** **BSc (Hons) Information & Communication Technologies** **BSc (Hons) Information Systems**		
MODULE NAME	**Web Technologies**		
DATE	**Wednesday 26 May 2010**	**MODULE CODE**	**CSE 2003Y(3)**
TIME	**9.30 – 12.30 Hrs**	**DURATION**	**3 Hours**
NO. OF QUESTIONS SET	**4**	**NO. OF QUESTIONS TO BE ATTEMPTED**	**4**

INSTRUCTIONS TO CANDIDATES

There are 4 questions in this paper.

All questions carry equal marks.

Answer all questions.

All questions refer to the Feedback case study discussed in class.

All questions will relate to the following feedback application that is being developed to collect feedback from students about different modules. The following schema has been defined for the feedback database, implemented in MySQL.

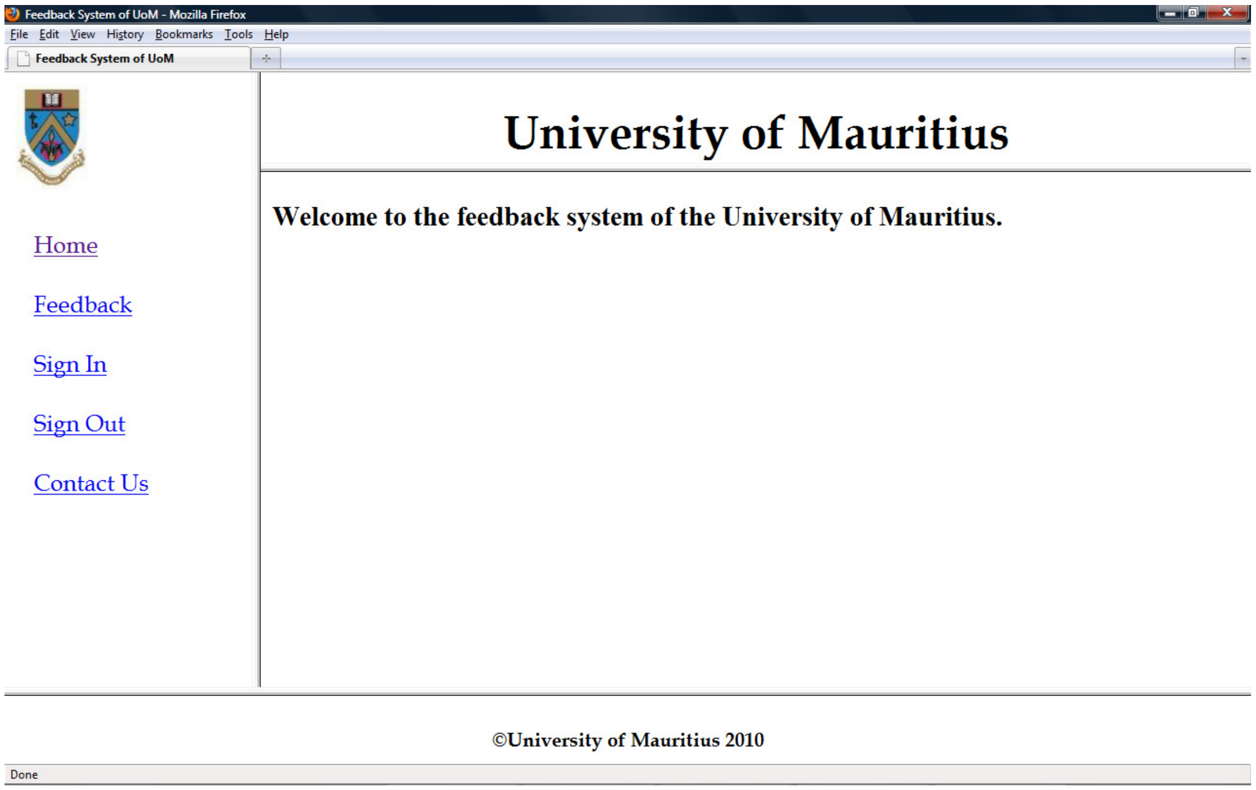
classsizes (classsize, classsizedesc)
feedbacks (email, modulecode, moduleyear, classsize, delivery, labs, Othercomments, moderation)
modules(modulecode, moduledesc)
users(username, pass)

Notes:

- *classsize* and *modulecode* in the *feedbacks* table are foreign keys from the classsizes and *modules* tables respectively.
- The *moderation* field (of type enum) contains values 'a' for approved, 'p' for pending, and 'r' for rejected. The default value is 'p'.
- The values for the *classsizes* table are '-1', '1' , '2' , '3' in the *classsize* field for 'poor', 'adequate', 'good' and 'excellent' respectively in the *classsizedesc* field.
- Delivery and labs fields are boolean and indicate whether improvements are required in the delivery and labs (a value of 1 indicates that improvement is required)

Question 1 (25 marks)

Consider the following page, **index.html**, shown in Figure 1, which is the home page for gathering feedback from students. The page is made up of four frames, menu, top, main and copyright loaded initially with the pages **menu.html**, **banner.html**, **main.html** and **copyright.html** respectively.


Feedback System of UoM - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Feedback System of UoM
Done

University of Mauritius

Welcome to the feedback system of the University of Mauritius.

Home

Feedback

Sign In

Sign Out

Contact Us

©University of Mauritius 2010

Figure 1: index.html

a) Write the codes for **index.html**. **[5 marks]**

b) The **menu.html** page contains 5 links: Home, Feedback, Sign In, Sign Out and Contact Us. Write the HTML codes for **menu.html** given that the links open **main.html**, **feedback.html**, **login.html** and **logout.html** respectively in the main frame. **Contact Us** opens a new email in the default email software with the address **feedback@uom.ac.mu** inserted automatically. **[5 marks]**

c) In the above example, both the links **Sign In** and **Sign Out** are displayed. Ideally if the user is logged in, only **Sign Out** should be displayed and if the user is not logged in, only **Sign In** should be displayed.
Modify the codes of **menu.html** such that the above is implemented, indicating any assumptions that you make. **[5 marks]**

…/Cont'd next page

Question 1 ***(Cont'd)***

d) Are there any changes to be made to the **index.html** page with respect to changes required in (c)? Explain your answer. **[2 marks]**

e) The URL for the above page is http://localhost:8086/webtech/exam/paper.html. What information is provided in the URL? **[2 marks]**

f) All the pages in **index.html** have been formatted using an external style sheet, **style.css**, found in the **style** subfolder. All the text and all the headings have been formatted using the font 'Book Antiqua'. All the links have been formatted such that normal link appears red, visited link appears green and active link appears blue.

- Write the codes for **style.css**.
- Write the codes to link the various pages to the external style sheet.
- Where is this code placed?

[4+1+1 marks]

Question 2

(a) To make the website more user-friendly, a **personalize website** link is provided in **menu.html**. When the user clicks on the link, the function **changeColor()** is called.

- The function makes a prompt appear asking the user for the background colour. If the colour is valid, the background colour of all four frames are changed to that colour. If the colour is invalid, the user is prompted for a valid value until a valid colour is entered.
- the user may enter the colour as an English word (red, blue etc) which is validated by the function validateAlphabet()
- the user may also enter the colour in the hexadecimal format (a # followed by 6 digits which can be a number from 0 to 9 or an alphabet between a to f) which is validated by the function validateHexa().
- If the user leaves the colour blank, he is prompted to enter a valid colour.
- Note that to refer to a frame in JavaScript, **top.framename** is used.
- Note that you are not required to use regular expressions.

i. Write the function **validateAlphabet()** in JavaScript. **[5 marks]**

ii. Write the function **validateHexa()** in JavaScript. **[7 marks]**

iii. Write the function **changeColor()** in JavaScript that makes use of the above 2 functions. **[7 marks]**

iv. How is the function **changeColor()** called and where? **[3 marks]**

.../Cont'd next page

Question 2 *(Cont'd)*

(b) Consider the following code. Predict the result after opening this page with the Internet Explorer browser and clicking on the *Add Module* button

```
<html>
<head>
<script type="text/javascript">
function addtablerow()
{
  var module_code=prompt("Enter Module Code");
  var module_name=prompt("Enter Module Name for " + module_code);
  table_modules.innerHTML=table_modules.innerHTML+"<tr><td
    width=200>"+module_code+"</td><td width=200>"+module_name+" </td></tr>";
  return false;
}

</script>
</head>
<body>
 <form action=addmodule_pro.php method=post onsubmit="return
       confirm('submit data')">
 <table id=table_modules>
  <tr>
    <th width=200>Module Code</th>
    <th width=200>Module Name</th>
  </tr>
 </table>
 <br><br>
 <button onclick="return addtablerow()">Add Module</button><input
     type=submit value=submit>
 </form>
</body>
</html>
```

[3 marks]

Question 3

Consider the page *addmodule.html*, as discussed in class, which is used to add modules to the database. The user is prompted for a *modulecode* and *modulename*, and has to enter valid *modulecodes* and *modulenames*. The codes for the page are as follows.

Code Listing: Addmodule.html

```
<html>
<head>
<script type="text/javascript">
<!--
 var count=0;  //global variable
 function addtablerow()
 {
  var module_code;
  var module_name;

  while (!(module_code=prompt("Enter Module Code")));

  while(!(module_name=prompt("Enter Module Name for " + module_code)));

  table_modules.innerHTML=table_modules.innerHTML+"<tr><td width=200><input
    type=text value="+module_code+" name=txt_modulecode"+ count +"></td><td
    width=200><input type=text value="+module_name+" name=txt_modulename"+
    count +"></td></tr>";

  count++;
  return false;
 }
-->
</script>
</head>
 <body>
 <form action=addmodule_pro.php method=post onsubmit="return confirm('submit
data')">
  <table id=table_modules>
    <tr>
     <th width=200>Module Code</th>
     <th width=200>Module Name</th>
    </tr>
  </table>
  <br><br>
  <button onclick="return addtablerow()">Add Module</button><input
       type=submit value=submit>
 </form>
</body>
</html>
```

One problem with the page is that multi-word values (words separated by space) in either the *modulecode* or the *modulename* are ignored and only the first word is passed to the input text boxes.

.../Cont'd next page

Question 3 *(Cont'd)*

You propose to change the codes highlighted in italics above as follows:

```
table_modules.innerHTML=table_modules.innerHTML+"<tr><td width=200><input
type=text value="+ module_code.replace(/\s/, "&nbsp;")+ "
name=txt_modulecode"+ count +"></td><td width=200><input type=text
value="+module_name.replace(/\s/,"&nbsp;")+" name=txt_modulename"+ count
+"></td></tr>";
```

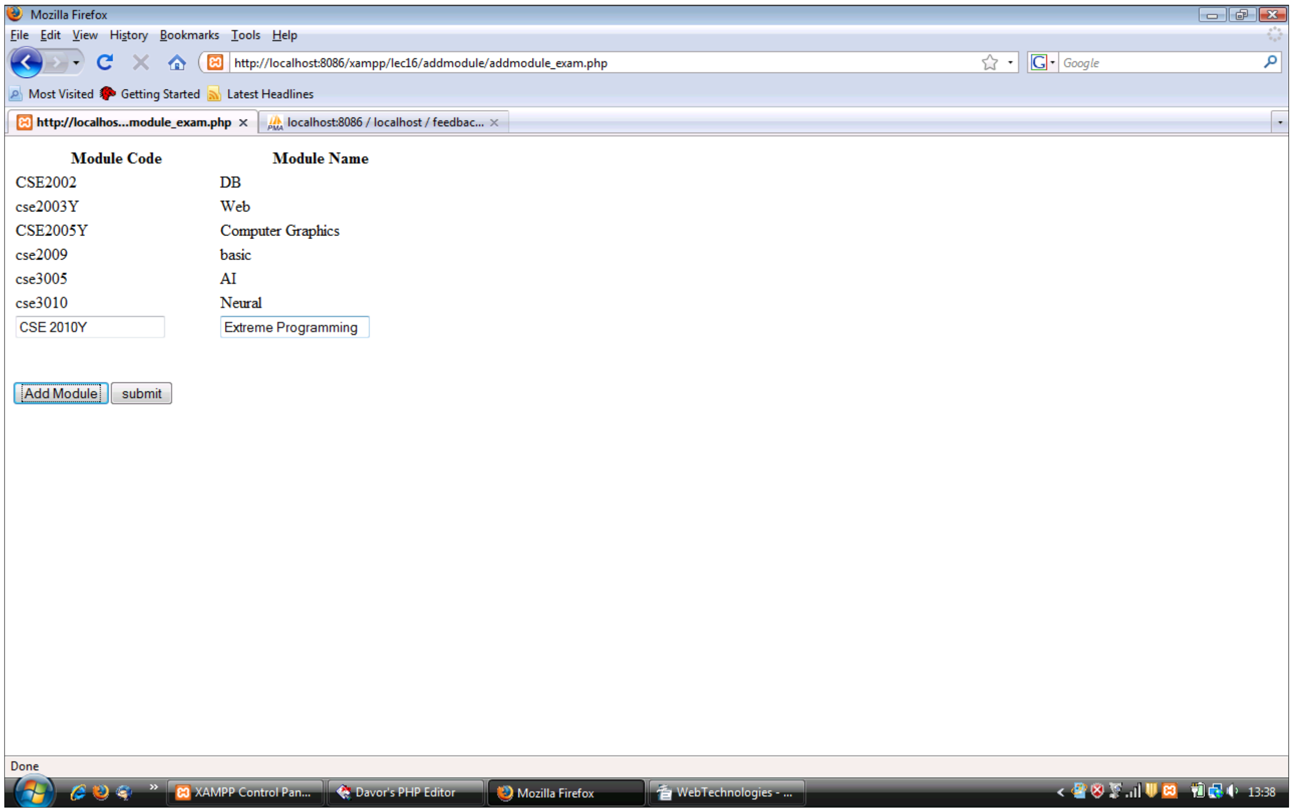
a) Explain the changes and why they work. **[3 marks]**

b) Another problem with the codes is that the codes only work in Firefox browser and not in the Internet Explorer browser. Briefly explain why. **[2 marks]**

c) The codes for *addmodule_pro.php* are as follows:

```
<?
$modulecount=0;
$modulecode="";
$modulename="";
include("db_connect.php");
foreach ($_POST as $key =>$value){
 if (substr($key,0,14)=="txt_modulecode")
 {

  $modulecount=substr($key,14,strlen($key));
  $modulecode=$value;
 }
 if (substr($key,0,14)=="txt_modulename")
 {

   if (substr($key,14,strlen($key))==$modulecount)
   //we have found the corresponding module name for the module
   {
     $modulename=$value;
     $sql_select="SELECT * FROM modules WHERE modulecode='$modulecode'";
     $Rs = mysql_query($sql_select);

     if (mysql_num_rows($Rs)<1)
     //insert only if the data has not been inserted before
     {
       //we want to insert data

        $sql_insert="INSERT  INTO  modules  (modulecode,  moduledesc)  values
('$modulecode','$modulename')";

        if (!mysql_query($sql_insert,$con))
        {
          die('Error: ' . mysql_error());
        }
     }
   }
 }//end if
}//end foreach
mysql_close($con)
?>
<html>
 <head>
   <title>Module_pro</title>
 </head>
 <body>
  Modules have been saved
 </body>
</html>
```

As illustrated in the codes, validations are made for modules that already exist in the database before inserting modules. However, the users of the system want to see all existing modules from the database on the *addmodule* page, so they do not waste time typing data for existing modules. A screen shot of the desired behavior is shown as follows:


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:8086/xampp/lec16/addmodule/addmodule_exam.php
Add Module
submit
Done

Module Code	Module Name
CSE2002	DB
cse2003Y	Web
CSE2005Y	Computer Graphics
cse2009	basic
cse3005	AI
cse3010	Neural
CSE 2010Y	Extreme Programming

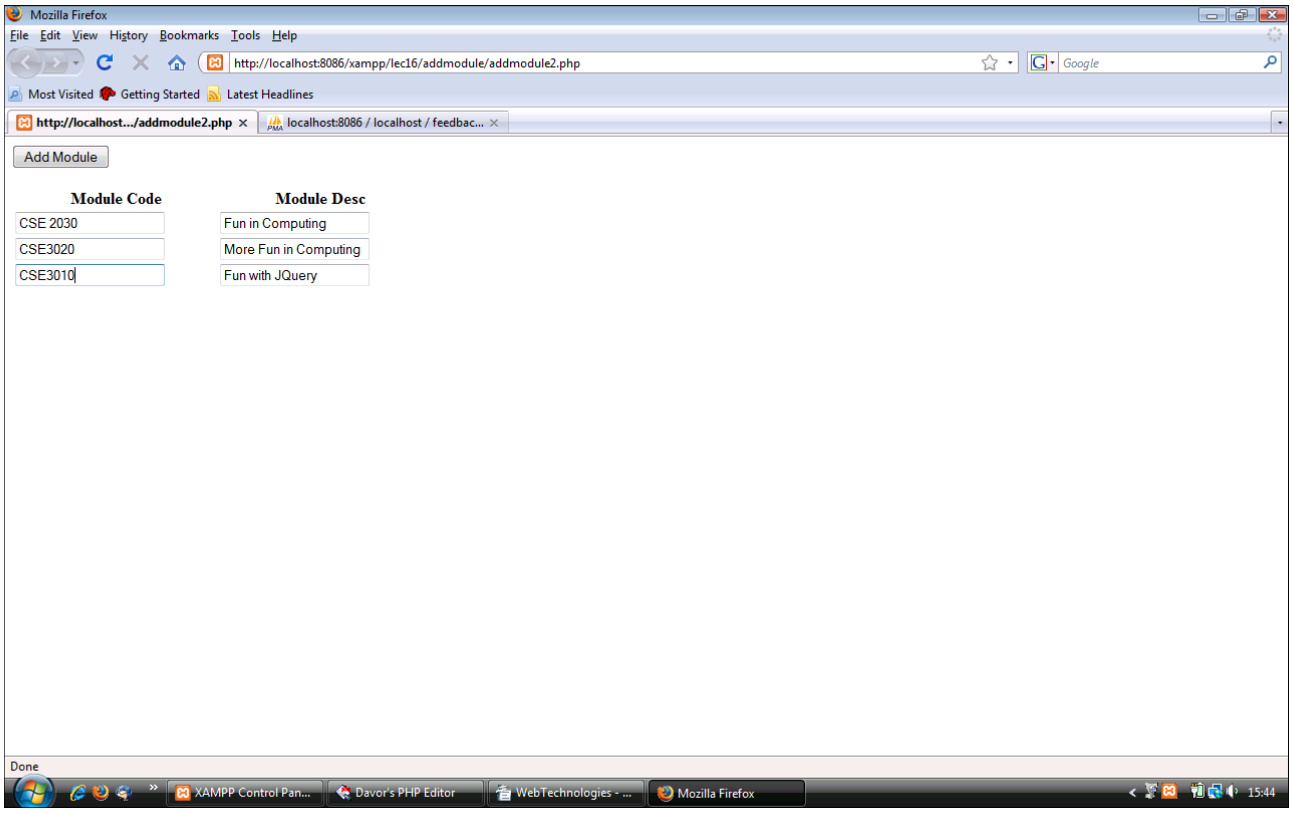
Figure2: addmodule.php

You decide to convert *addmodule.html* to *addmodule.php* and add the new functionality. Write the codes to take care of the above requirements. Assume you have a file, *db_connect.php* that connects to the database. **[10 marks]**

.../Cont'd next page

Question 3 *(Cont'd)*

Another problem with the *addmodule.html* page is that all the data is being saved locally until finally submitted to the server. Many times, the user session expires and the user loses all her work. In this respect, you decide to implement the functionality in AJAX, such that whenever the user has been prompted for the *modulecode* and *modulename* values, these are submitted to the server asynchronously, and if the server returns *success*, the rows are added to be displayed to the bottom of the page. A sample screenshot for *addmoduleajax.html* is as shown in Figure 3:


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:8086/xampp/lec16/addmodule/addmodule2.php
Add Module
Done

Module Code	Module Desc
CSE 2030	Fun in Computing
CSE3020	More Fun in Computing
CSE3010	Fun with JQuery

Figure 3: addmoduleajax.html

…/Cont'd next page

Question 3 *(Cont'd)*

Part of the codes for the page *addmoduleajax.html* are as shown:

```
  var http_request;
  var module_code;
  var module_name;
  function makePOSTRequest(url, parameters) {


   http_request = false;
      if (window.XMLHttpRequest) { // Mozilla, Safari,...
         http_request = new XMLHttpRequest();
         if (http_request.overrideMimeType) {
                // set type accordingly to anticipated content type
            //http_request.overrideMimeType('text/xml');
            http_request.overrideMimeType('text/html');
         }
      } else if (window.ActiveXObject) { // IE
         try {
            http_request = new ActiveXObject("Msxml2.XMLHTTP");
         } catch (e) {
            try {
               http_request = new ActiveXObject("Microsoft.XMLHTTP");
            } catch (e) {}
         }
      }
      if (!http_request) {
         alert('Cannot create XMLHTTP instance');
         return false;
      }

      http_request.onreadystatechange = addModule;
      http_request.open('POST', url, true);
      http_request.setRequestHeader("Content-type",   "application/x-www-form-
urlencoded");
      http_request.setRequestHeader("Content-length", parameters.length);
      http_request.setRequestHeader("Connection", "close");
      http_request.send(parameters);
   }


 function addTableRow(obj) {

  while (!(module_code=prompt("Enter Module Code")));

  while(!(module_name=prompt("Enter Module Name for " + module_code)));
 var   poststr   =   "txt_modulecode="   +   escape(encodeURI(module_code))   +
"&txt_modulename=" + escape(encodeURI(module_name));

 makePOSTRequest('addmodule_pro1.php', poststr);
 }
```

Write the codes for the *addModule()* function. **[10 marks]**

Question 4:

Recall the example on XML DOM as discussed in class related to the same functionality in Question 3 above. The codes are as follows:

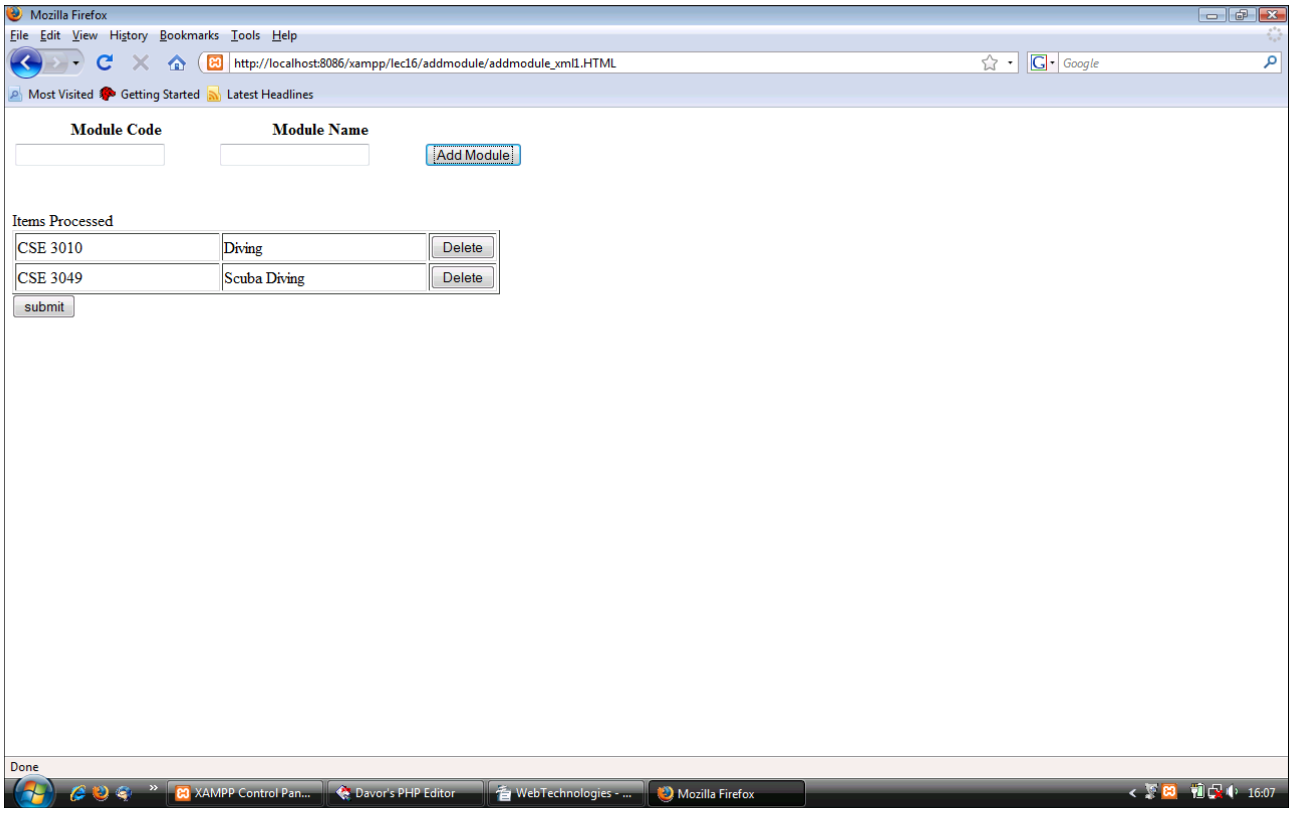
```
<html>
<head>
<script type="text/javascript">
<!--
  var count_modules=0;
  var xml_modules="";
   xml_modules=
       "<modules><module><modulecode></modulecode><moduledesc></moduledesc>
            </module></modules>";

  var xmlDoc_modules = loadXMLString(xml_modules);

  function loadXMLString(txt)
  {
    var xmlDoc;

    try //Internet Explorer
    {
      xmlDoc=new ActiveXObject("Microsoft.XMLDOM");
      xmlDoc.async="false";
      xmlDoc.loadXML(txt);
      //xmlDoc.encoding="UTF-8";
      return xmlDoc;
    }
    catch(e)
    {
      try //Firefox, Mozilla, Opera, etc.
      {
        parser=new DOMParser();
        xmlDoc=parser.parseFromString(txt,"text/xml");
        xmlDoc.encoding="UTF-8";
        return xmlDoc;
      }
      catch(e) {alert(e.message)}
    }
    return(null);
  }//end function loadXMLString

  function insertAddedDescription(modulecode, moduledesc)
  {
     modules_node=xmlDoc_modules.childNodes[0];
     len = modules_node.childNodes.length;
     //fill in the empty nodes
     modulecode_node = modules_node.childNodes[len-1].childNodes[0];
     modulecode_node_val = xmlDoc_modules.createTextNode(modulecode);
     modulecode_node.appendChild(modulecode_node_val);

     moduledesc_node = modules_node.childNodes[len-1].childNodes[1];
     moduledesc_node_val = xmlDoc_modules.createTextNode(moduledesc);
     moduledesc_node.appendChild(moduledesc_node_val);

     //create the new empty replacing node
     module_node = xmlDoc_modules.createElement("module");
     modulecode_node =xmlDoc_modules.createElement("modulecode");
     moduledesc_node =xmlDoc_modules.createElement("moduledesc");
     module_node.appendChild(modulecode_node);
     module_node.appendChild(moduledesc_node);
     modules_node.appendChild(module_node);
     return true;
  }


  function addtablerow()
  {
    var modulecode=document.forms[0].txt_modulecode.value;
    var moduledesc = document.forms[0].txt_moduledesc.value;
    insertAddedDescription(modulecode, moduledesc);
    document.forms[0].txt_modulecode.value="";
    document.forms[0].txt_moduledesc.value="";
    //recompute display
    displayItems()
    return false; //to prevent the data from being submitted to the server
  }//end function addtablerow()

  function displayItems()
  {
     table_str= "Items Processed <br/><table border=1>";
     modules_div = document.getElementById("div_modules");
     modules_node=xmlDoc_modules.childNodes[0];
     len = modules_node.childNodes.length;

     for (i=0; i<len-1 ;i++)
     {
       table_str = table_str + "<tr><td width=200>";
       module_code =
           modules_node.childNodes[i].childNodes[0].firstChild.nodeValue;
       table_str = table_str + module_code +"</td><td width=200>";
       module_desc =
            modules_node.childNodes[i].childNodes[1].firstChild.nodeValue;
       table_str = table_str + module_desc +"</td></tr>";
     }
     table_str = table_str + "</table>";
     modules_div.innerHTML = table_str;

  }//end function displayItems()

  function encode_Items()
  {
   var browser=getBrowser();
   var xmlstring;
   if (browser=="ie")
   {
     xmlstring=xmlDoc_modules.xml;
     document.forms[0].txt_xml_modules.value= xmlstring;
     return true;
   }
   if (browser=="ff")
   {
     xmlstring = (new XMLSerializer()).serializeToString(xmlDoc_modules);

     document.forms[0].txt_xml_modules.value= xmlstring;
  return true;
   }
   return false;
  }//end function encode_Items()

  function getBrowser()
  {
    var sbrowser=navigator.userAgent;
    if (sbrowser.toLowerCase().indexOf('msie')>-1)
      return "ie";
    if (sbrowser.toLowerCase().indexOf('firefox')>-1)
      return "ff";
    return "";
  }//end function getBrowser()



-->
</script>
</head>
 <body>
 <form action=addmodulexml_pro.php method=post onsubmit="return
encode_Items()">
  <table id=table_modules>
    <tr>
     <th width=200>Module Code</th>
     <th width=200>Module Name</th>
     <th>
    </tr>
    <tr>
     <td><input type=text name=txt_modulecode></td>
     <td><input type=text name=txt_moduledesc></td>
     <td><button onclick="return addtablerow()">Add Module</button></td>
    </tr>
  </table>
  <br><br>
  <div id="div_modules"></div>
  <input type=submit value=submit onclick='return encode_Items()'>
  <input type=hidden name=txt_xml_modules id=txt_xml_modules>
 </form>
</body>
</html>
```

These codes allowed the user to enter the modulecodes and modulesnames, which are stored in an XML file, and finally, the XML file is sent to the server.

One problem with the above codes is that they do not allow the user to delete any of the entries made, and you have been asked to change the codes such that the user can delete any row he wants to, before the data is submitted to the server. A screenshot of the system should be as follows:

…/Cont'd next page

Question 4 *(Cont'd)*



Module Code Module Name

Items Processed

CSE 3010	Diving	Delete
CSE 3049	Scuba Diving	Delete

Figure 4: addmodulexml.html

a) When the delete button is clicked, the appropriate node is deleted from the XML tree and the display is refreshed.

Write the codes for the above.

Ensure that you have the codes for:

- Adding the Delete button for each row that is added and associating it with the proper function such that the proper node is deleted from the XML tree.
- Refreshing the display accordingly after a successful delete (i.e. after the node has been removed from the XML tree).

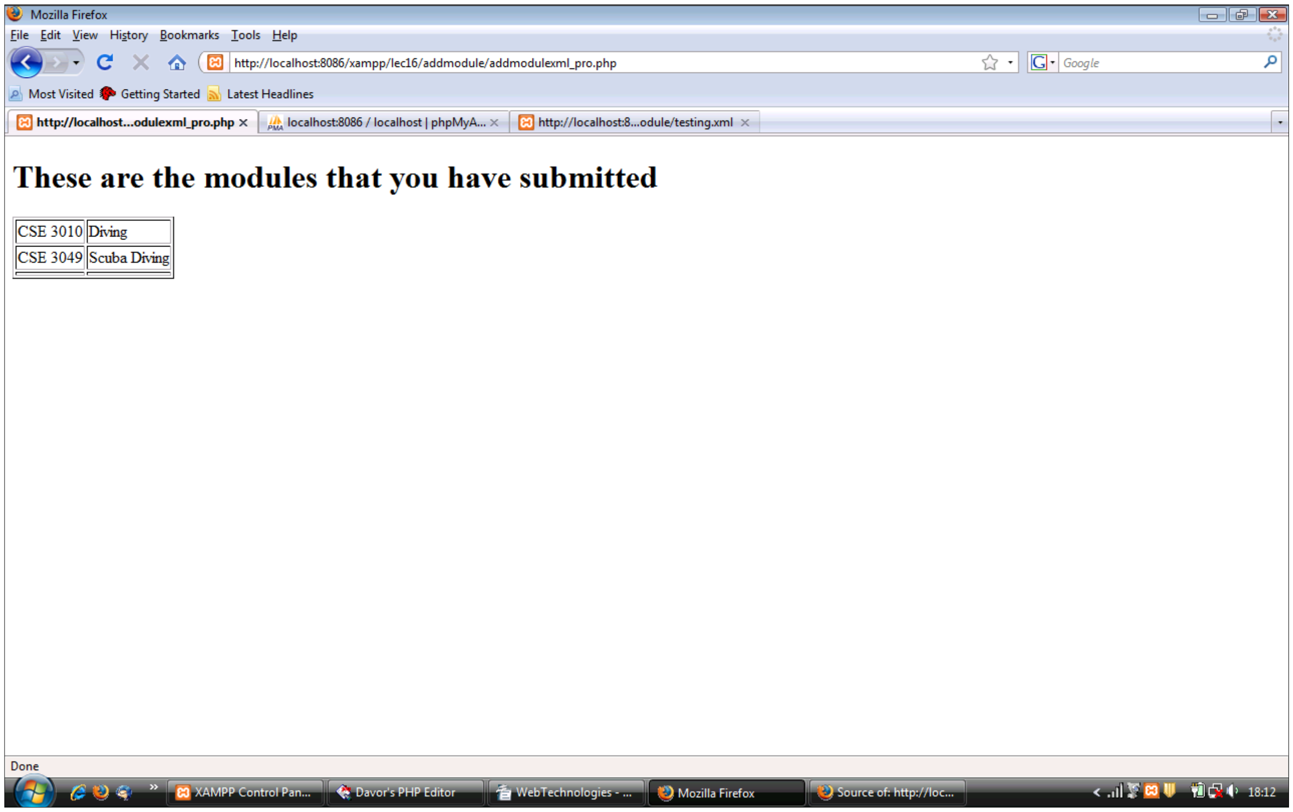
Note:You may find the *DOMelement.removeChild(node)* method which removes *node* from *DOMelement useful.* **[15 marks]**

…/Cont'd next page

Question 4 *(Cont'd)*

b) Write the schema, *modules.xsd,* for validating the XML that is submitted to the server. **[5 marks]**

c) The page above is submitted to *addmodulexml_pro.php* and is processed by the *modules.xsl* XSLT, such that the display is as follows:


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:8086/xampp/lec16/addmodule/addmodulexml_pro.php
Most Visited Getting Started Latest Headlines
http://localhost...odulexml_pro.php
localhost:8086 / localhost | phpMyA...
http://localhost:8...odule/testing.xml
Done

These are the modules that you have submitted

CSE 3010	Diving
CSE 3049	Scuba Diving

Figure 5: addmodulexml_pro.php

The codes for *addmodulexml_pro.php* is shown below for reference purpose.

Listing for addmodulexml_pro.php

```
<?
 $modulecode="";
 $moduledesc="";

 $modules= $_POST['txt_xml_modules'];

 $xml_Doc = new DOMDocument();
 $xml_Doc->loadXML($modules);

 if(!$xml_Doc->schemaValidate('modules.xsd'))
   echo "Your xml is NOT valid";
```

```
else
 {
    include("db_connect.php");
    header('Content-Type: text/xml');

    $xml_modules=simplexml_import_dom($xml_Doc);


    foreach ($xml_modules->module as $module)
    {
      $modulecode=$module->modulecode;
      $moduledesc= $module->moduledesc;

      if($modulecode!="" && $modulecode!=null)
      {
       $sql_select="SELECT * FROM modules WHERE modulecode='$modulecode'";
       $Rs = mysql_query($sql_select);

       if (mysql_num_rows($Rs)<1)
       //insert only if the data has not been inserted before
       {
       //we want to insert data

         $sql_insert="INSERT  INTO  modules  (modulecode,  moduledesc)  values
('$modulecode','$moduledesc')";

         if (!mysql_query($sql_insert,$con))
         {
           die('Error: ' . mysql_error());
         }
       }//end if
      }//end if
    }//end foreach

   //now build an XML to display the modules.
   $xml_output = "<?xml version=\"1.0\" encoding=\"ISO-8859-1\"?>";
   $xml_output=$xml_output."<?xml-stylesheet type=\"text/xsl\"
        href=\"modules.xsl\"?>";
   $xml_output = $xml_output.$modules;
   echo $xml_output;

   mysql_close($con);
  }//end else
?>
```

Write the codes for *modules.xsl*. **[5 marks]**

END OF QUESTION PAPER

/ph