

UNIVERSITY OF MAURITIUS

FACULTY OF ENGINEERING



SECOND SEMESTER EXAMINATIONS

MAY 2011

PROGRAMME	BSc (Hons) Computer Applications (Full-Time)		
MODULE NAME	Web Technologies II		
DATE	Tuesday 17 May 2011	MODULE CODE	CSE 2041(3)
TIME	13:30 –15:30 Hrs	DURATION	2 hours
NO. OF QUESTIONS SET	4	NO. OF QUESTIONS TO BE ATTEMPTED	4

INSTRUCTIONS TO CANDIDATES

Answer ALL questions.

All questions carry equal marks.

Consider a feedback application that is being developed to collect feedback from lecturers about students' performance. Each class is divided into 2 batches. For example, Computer Applications of Year 2009 is divided into CA Group A+B and CA Group C. The following schema has been defined for the feedback database, implemented in MySQL.

Batch (BatchId, BatchName, Year, BatchDesc, NoOfStudents, Repeat)

Possible values (1, 'CA Group C', 2009, 'Computer Applications Group C', 25, 0)

feedbacks (username, BatchId, modulecode, attendance, participation, satisfaction)

Possible values('anwar.c', 2, 'CSE2041', 2, 3, 2)

modules(modulecode, modulename, moduletype)

Possible values('CSE2041', 'Web Technology II', 0)

lecturer(username, Name, Surname, pass)

Possible values('anwar.c', 'Anwar', 'Chuttoo', 'secret')

Notes:

feedbacks table:

username, *BatchId* and *modulecode* in the *feedbacks* table are foreign keys from the *lecturer*, *Batch* and *modules* tables respectively.

The possible values for the *attendance* and *participation* fields are '0', '1', '2', '3' denoting 'poor', 'adequate', 'good' and 'excellent' respectively.

The possible values for the *satisfaction* field are '-3', '0', '1', '5' for 'Very unsatisfied', 'Fair', 'Satisfied' and 'excellent' respectively.

Batch table:

BatchDesc is a description of the batch. For example 'Computer Applications, Groups A and B' for *BatchName* field value 'CA Group A+B'.

Year denotes the year the batch of students joined UoM.

NoOfStudents denotes the number of students in the batch

Repeat is a Boolean value to denote whether this batch is a repeater batch. '1' for repeater and '0' for non-repeater.

modules table:

The possible values for the *moduletype* field are '0', '1', '2', '3' for 'CORE', 'Elective', 'DE Manual', 'GEM' respectively.

SECTION A**Question 1 - [Total 25 Marks]**

Before a lecturer can input feedback about a particular batch he/she has serviced, there is a need to have the required modules registered on the system. The lecturer has to successfully login first, then he/she would be presented with the **Add Modules** page (Figure 1: *addmodules.php*) to enter the modules' details.

To register a new module, the user has to enter the required data and press the 'Check in Database' button. If the module code already exists, the system displays the error message *'* This module already exists!'*, else it adds the data to the *New Modules* table.

Add Modules

User: anwar.c

Module Code:

Module Name:

Module Type:

New Modules

Module Code	Module Name	Module Type
CSE2142	Software Engineering	0
CSE3034	Graphics Design	1

Figure 1: *addmodules.php*

Given an extract of the **addmodules.php** page (Figure 2)

```
<tr><td align="center" colspan="4">
<input type="button" id="btnSend" name="btnSend" value="Check in Database"
onclick="sendRequestValid('/ExWeb2May2011/isvalid.php?','txtModuleCode',txtModuleCode.value)"/>
</td></tr>
```

Figure 2: *addmodules.php* extract**ADDITIONAL INFORMATION:**

- txtModuleCode refer to the first textbox in **addmodules.php** page.

Question 1 [Cont'd]

Code listings for the **Check in Database** process (moduleValidate.js) (Figure 3)

```
function createXHR()
{
    try { return new XMLHttpRequest(); } catch(e) {}
    // older IE Browser compatibility check - Not required to implement
}

function sendRequestValid(url,elementId,elementvalue)
{
    var params = elementId + "=" + elementvalue;
    var xhrValid = createXHR();
    if (xhrValid)
    {
        // 1 - MISSING CODES - Calls handleResponseValid(xhrValid)
    }
}

function handleResponseValid(xhrValid)
{
    if (// 2 - MISSING CODES - Test Response)
    {
        // 3 - MISSING CODES - Process returned XML file
        if(xmlRoot.firstChild.nodeValue == "SUCCESS")
        {
            document.getElementById("result").innerHTML = "";
            var table = document.getElementById("tblAddModules");
            // 4 - MISSING CODES - Adds validated modules to the New Modules table
        }
        else
        {
            document.getElementById("result").innerHTML = "* This Module already exists!";
        }
    }
}
```

Figure 3: modulevalidate.js

- (a) You are required to fill in the missing codes for Figure 3. Ensure your code works in both IE and Firefox and the request is sent via **post**.

[5+3+3+4 = 15 Marks]

ADDITIONAL INFORMATION:

- The *handleResponseValid(xhrValid)* function receives an XML file constructed as **<message>SUCCESS</message>** if module is not found in database. Else **<message>FAIL</message>** if the module already exists.
- tblAddModules** is the *id* of the New Modules table in **addmodules.php**.

- (b) Write the codes for the **isvalid.php** file. You can use `$_REQUEST` to capture data from the form.

[6 Marks]

ADDITIONAL INFORMATION:

- sendRequestValid* function is sending data to the **isvalid.php** (Figure 2, 3)
- handleResponseValid* function receives an XML file from the same php file.
- Make sure you do not allow caching of data on the server.
- add a reference to **dbconnect.php**.
- The following sql code may help:

.../Cont'd next page

Question 1 [Cont'd]

```
$sql = "SELECT * FROM modules where modulecode='".$mCode.'";
```

On clicking on the *submit* button (Figure 1), the page constructs an XML file to send to a PHP file for inserting the items in the *modules* table in the *feedback2011* database. Part of the structure of the XML document is given below (Figure 4).

- (c) You are required to complete the XML file (Figure 4) based on data shown in Figure 1.

```
<!-- 1 – Missing Codes -->
<modules>
  <module modulecode='CSE2142'>
    <!-- 2 – Missing Codes -->
  </module>
  <!-- 3 – Missing Codes -->
</modules>
```

Figure 4

[1+1+2 = 4 Marks]

ADDITIONAL INFORMATION:

- The root element is **modules**
- The **modulecode** is always an attribute of the **module** element
- **modulename** and **moduletype** are elements

Question 2 - [Total 25 Marks]

Lecturers can check the feedback entered for serviced batches at any time. The form below (Figure 5: *feedbacks.php*) is used to query the database for the performance of the batches listed in the *Queries* table.

Performance Feedback Query

User: anwar.c

Batch: CS Batch A+B 2009 Rep Module: CSE3032 - Semantic Web Add Row

Queries

Batch	Module Code
CA Batch C 2009	CSE2041
CS Batch A+B 2009 Rep	CSE3032

Submit

Figure 5: *feedbacks.php*

The user selects the batch and module from their respective drop down list and click on the *Add Row* button to add the information to the *Queries* table.

On clicking the submit button, the following XML string (Figure 6) which was stored in a hidden field, *hfXML* (found in *feedbacks.php*), is sent to the *computeTotal.php* page via the *POST* method.

```
<queries>
  <user>anwar.c</user>
  <batches>
    <batch>
      <name id="2">CA Batch C 2009</name>
      <modulecode>CSE2041</modulecode>
    </batch>
    <batch>
      <name id="3">CS Batch A+B 2009 Rep</name>
      <modulecode>CSE3032</modulecode>
    </batch>
  </batches>
</queries>
```

Figure 6

.../Cont'd next page

Question 2 - [Cont'd]

Part of the *computeTotal.php* file is given below (Figure 7).

```
<?php
    header('Content-Type: text/xml');
    if(isset($_POST['hdnXML'])){
        $strXML = $_POST['hdnXML'];
    }else{
        $strXML = "<root>error</root>";
    }
    // 1 - Missing codes for loading the XML string
    if(// 2 - Missing codes for validating against the XSD schema definition file){
        echo "Your XML is not valid";
    }
    else{
        include($_SERVER['DOCUMENT_ROOT'].'/ExWeb2May2011/dbconnect.php');
        // Missing codes for
        // 3 - using the XML string and query the database for performance
        // 4 - construct another XML string for displaying Performance

        $xmlOutput = "<?xml version='1.0' encoding='UTF-8'?">";

        // 5 - Missing codes for adding
        // - the constructed feedback XML string
        // - adding the stylesheet

        echo $xmlOutput;
        mysql_close($conn);
    }//end else
?>
```

Figure 7: *computeTotal.php*

- (a) You are required to fill in the missing codes for *computeTotal.php*, ensuring the following:
- (i) Load the XML string into a DOM object
 - (ii) Validate the XML string given in Figure 6 against a XML Schema Definition (XSD) file named ***batch.xsd*** found in the same directory as the current page.
 - (iii) Query the *feedback2011* database in order to get the *username*, *FullBatchName*, *modulecode* and *Performance* for each *batch* from the XML tree generated using XML data from Figure 6.

You may find the following codes useful:

```
$sql_select ="SELECT username, CONCAT(batch.BatchName,' ',batch.Year,' ',IF(batch.Repeat=0,'Rep')) AS FullBatchName,
modulecode,";
$sql_select = $sql_select." feedbacks.attendance+ feedbacks.participation+feedbacks.satisfaction AS Performance";
$sql_select = $sql_select." FROM feedbacks INNER JOIN batch ON feedbacks.BatchId = batch.BatchId";
$sql_select = $sql_select." WHERE username='".$Username.'" AND feedbacks.BatchId='".$batchId.'" AND
modulecode='".$ModuleCode.'"";
$rs = mysql_query($sql_select);
$rows = mysql_fetch_array($rs);
```

.../Cont'd next page

Question 2 - [Cont'd]

- (iv) Construct the feedback XML tree in the following format:

```
<?xml version="1.0" encoding="UTF-8"?>
<queries>
  <user>anwar.c</user>
  <batches>
    <batch id="2">
      <fullbatchname>CA Batch C 2009 </fullbatchname>
      <modulecode>CSE2041</modulecode>
      <performance>7</performance>
    </batch>
    <batch id="3">
      <fullbatchname>CS Batch A+B 2009 Rep</fullbatchname>
      <modulecode>CSE3032</modulecode>
      <performance>5</performance>
    </batch>
  </batches>
</queries>
```

- (v) Finally link the **batch.xsl** stylesheet definition found in the same directory as the current page to transform the feedback XML.

[2+2+7+6+2 = 19 Marks]

ADDITIONAL INFORMATION:

- The default namespace for the stylesheet is <http://www.w3.org/1999/XSL/Transform>
- (b) A programmer can also choose to use an XSLT processor to transform an XML. Assuming you are the programmer, write the codes for transforming the feedback XML string using the XSLT Processor and *batch.xsl* file.

ADDITIONAL INFORMATION:

- You are not required to write the whole *computeTotal.php* file; only the fragment of codes pertaining to getting the XML string, feeding it to the XSLT Processor and getting the output.

[6 Marks]

SECTION BQuestion 3 – [Total 25 Marks]

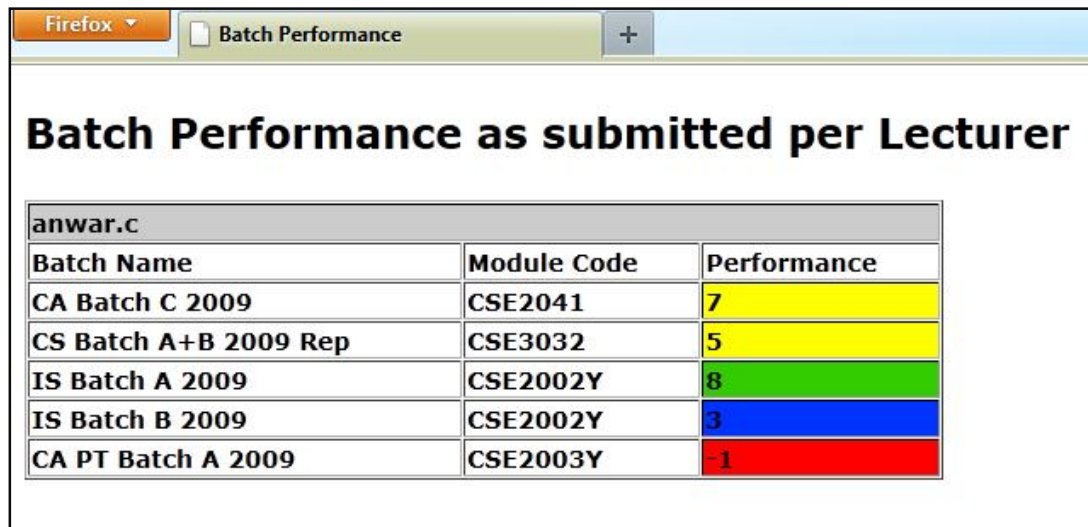
- (a) The **feedback XML tree** from **Question 2(a) (iv)** has been re-written for you below. Write down the schema definition, **batch.xsd** to validate the instance document taking into consideration the following:
- Each lecturer must have **at least one** batch with which he/she has worked with.
 - The maximum number of batches allowed per lecturer is **five**.
 - The element performance takes as default value **'0'**.
 - Each **'queries'** element consists of **only one** lecturer username.
 - Given that the **'batch'** element has more than occurrences and that each occurrence has the same set of child elements, create a special type with name **'batchType'** that will define the **'batch'** element property.
 - You may provide any other assumption you feel applicable.
 - The default namespace for the schema is:
<http://www.w3.org/2001/XMLSchema>

```
<?xml version="1.0" encoding="UTF-8"?>
<queries>
  <user>anwar.c</user>
  <batches>
    <batch id="2">
      <fullbatchname>CA Batch C 2009 </fullbatchname>
      <modulecode>CSE2041</modulecode>
      <performance>7</performance>
    </batch>
    <batch id="3">
      <fullbatchname>CS Batch A+B 2009 Rep</fullbatchname>
      <modulecode>CSE3032</modulecode>
      <performance>5</performance>
    </batch>
  </batches>
</queries>
```

[15 Marks]

- (b) The following page results from the transformation of the **feedback XML tree** using the **batch.xsl**.

.../Cont'd next page

Question 3 [Cont'd]


The screenshot shows a web browser window with the title 'Batch Performance'. The main heading is 'Batch Performance as submitted per Lecturer'. Below it is a table with the following data:

Batch Name	Module Code	Performance
CA Batch C 2009	CSE2041	7
CS Batch A+B 2009 Rep	CSE3032	5
IS Batch A 2009	CSE2002Y	8
IS Batch B 2009	CSE2002Y	3
CA PT Batch A 2009	CSE2003Y	-1

- (i) Write down the codes for **batch.xsl**, ensuring the following conditions be respected:

Condition	Background colour for Performance cell
Performance ≤ 0	RED
$1 \leq \text{Performance} \leq 3$	BLUE
$4 \leq \text{Performance} \leq 7$	YELLOW
$8 \leq \text{Performance} \leq 11$	GREEN

- (ii) The default namespace for the stylesheet is <http://www.w3.org/1999/XSL/Transform>

[10 marks]

Question 4 - [Total 25 marks]

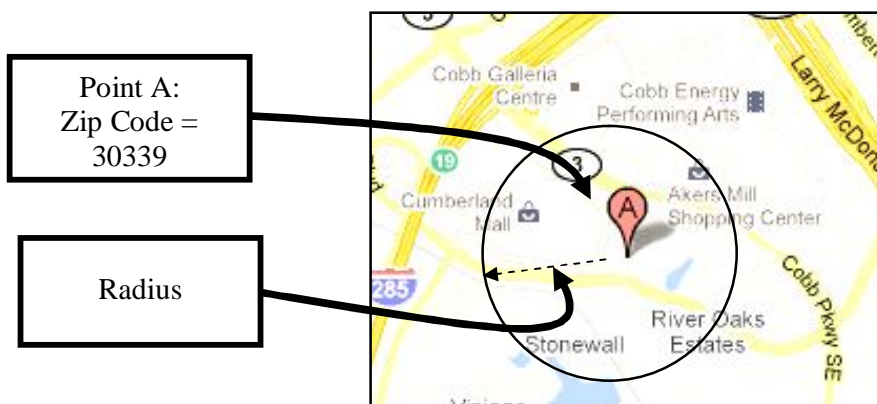
Consider the WSDL file available at: <http://www.ignyte.com/webservices/ignyte.whatsshowing.webservice/moviefunction.smx?wsdl> which is used to display movies that are being played in specified regions. The web service also provides information on upcoming films.

Use the WSDL file to create a client: **movie.php** that will pass on parameters such as zip code and radius and finally output the theatres and the movies they are playing in that specific region. A screenshot of the input and output screens are given below:

Input Screen:



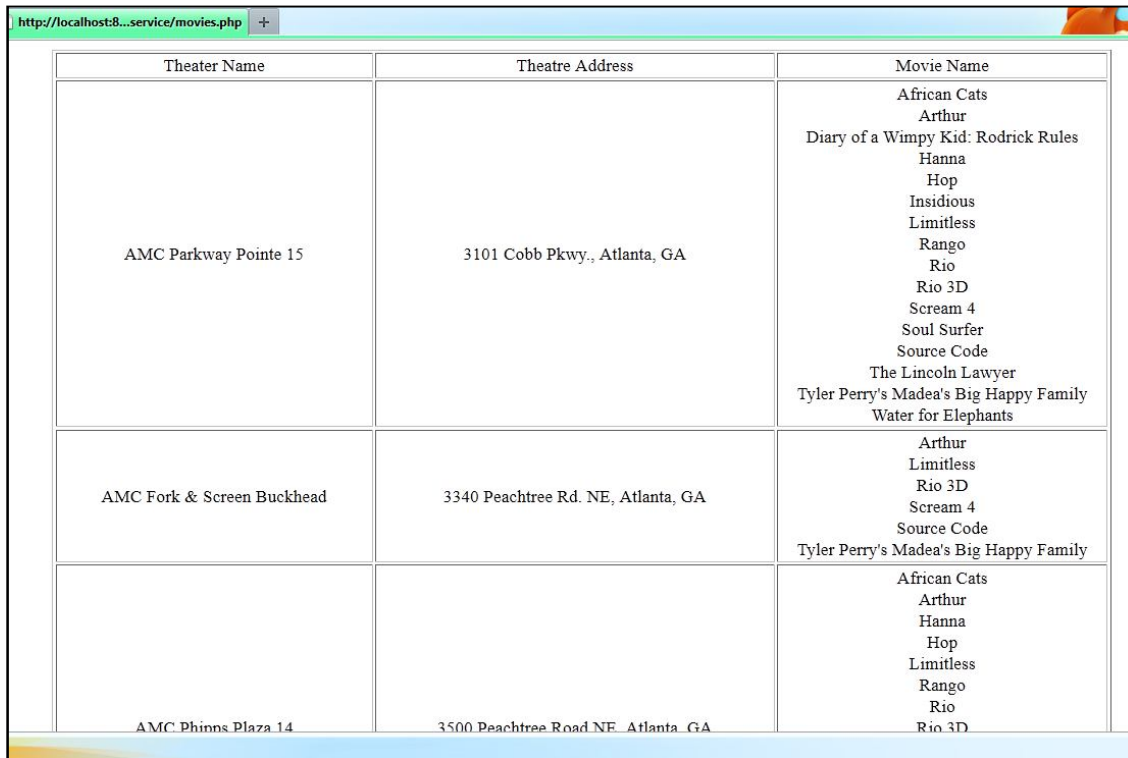
- The first textbox takes as input the zip code for the region where you want to see the available theatres and the movies they are playing.
 - Textbox name = zip_txt.
- The second textbox welcomes the value of radius, i.e. a value for distance all around the point specified by the zip code. The diagram below depicts this.
 - Textbox name = radius_txt.



.../Cont'd next page

Question 4 [Cont'd]

Output Screen:



Theater Name	Theatre Address	Movie Name
AMC Parkway Pointe 15	3101 Cobb Pkwy., Atlanta, GA	African Cats Arthur Diary of a Wimpy Kid: Rodrick Rules Hanna Hop Insidious Limitless Rango Rio Rio 3D Scream 4 Soul Surfer Source Code The Lincoln Lawyer Tyler Perry's Madea's Big Happy Family Water for Elephants
AMC Fork & Screen Buckhead	3340 Peachtree Rd. NE, Atlanta, GA	Arthur Limitless Rio 3D Scream 4 Source Code Tyler Perry's Madea's Big Happy Family
AMC Phinns Plaza 14	3500 Peachtree Road NE, Atlanta, GA	African Cats Arthur Hanna Hop Limitless Rango Rio Rio 3D

Notice that the result needed as output from the web service consists of only:

- The name for the theatres within the radius specified
- The address of those theatres
- The films being played by the theatres

The WSDL file is given on the following page.

.../Cont'd next page

Question 4 [Cont'd]

```

<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions
  targetNamespace="http://www.ignite.com/whatsshowing"
  xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
  xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
  xmlns:s="http://www.w3.org/2001/XMLSchema"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
  xmlns:tns="http://www.ignite.com/whatsshowing"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">

  <wsdl:types>

    <s:schema elementFormDefault="qualified"
      targetNamespace="http://www.ignite.com/whatsshowing">

      <s:element name="GetTheatersAndMovies">
        <s:complexType>
          <s:sequence>
            <s:element maxOccurs="1" minOccurs="0" name="zipCode" type="s:string" />
            <s:element maxOccurs="1" minOccurs="1" name="radius" type="s:int" />
          </s:sequence>
        </s:complexType>
      </s:element>

      <s:element name="GetTheatersAndMoviesResponse">
        <s:complexType>
          <s:sequence>
            <s:element maxOccurs="1" minOccurs="0"
              name="GetTheatersAndMoviesResult" type="tns:ArrayOfTheater" />
          </s:sequence>
        </s:complexType>
      </s:element>

      <s:complexType name="ArrayOfTheater">
        <s:sequence>
          <s:element maxOccurs="unbounded" minOccurs="0" name="Theater"
            nillable="true" type="tns:Theater" />
        </s:sequence>
      </s:complexType>

      <s:complexType name="Theater">
        <s:sequence>
          <s:element maxOccurs="1" minOccurs="0" name="Name" type="s:string" />
          <s:element maxOccurs="1" minOccurs="0" name="Address" type="s:string" />
          <s:element maxOccurs="1" minOccurs="0" name="Movies"
            type="tns:ArrayOfMovie" />
        </s:sequence>
      </s:complexType>
    </s:schema>
  </wsdl:types>

```

.../Cont'd next page

Question 4 [Cont'd]

```

<s:complexType name="ArrayOfMovie">
  <s:sequence>
    <s:element maxOccurs="unbounded" minOccurs="0" name="Movie"
              nillable="true" type="tns:Movie"/>
  </s:sequence>
</s:complexType>

<s:complexType name="Movie">
  <s:sequence>
    <s:element maxOccurs="1" minOccurs="0" name="Rating" type="s:string"/>
    <s:element maxOccurs="1" minOccurs="0" name="Name" type="s:string"/>
    <s:element maxOccurs="1" minOccurs="0" name="RunningTime"
              type="s:string"/>
    <s:element maxOccurs="1" minOccurs="0" name="ShowTimes"
              type="s:string"/>
  </s:sequence>
</s:complexType>
</s:schema>
</wsdl:types>

<wsdl:message name="GetTheatersAndMoviesSoapIn">
  <wsdl:part element="tns:GetTheatersAndMovies" name="parameters"/>
</wsdl:message>

<wsdl:message name="GetTheatersAndMoviesSoapOut">
  <wsdl:part element="tns:GetTheatersAndMoviesResponse" name="parameters"/>
</wsdl:message>

<wsdl:portType name="MovieInformationSoap">
  <wsdl:operation name="GetTheatersAndMovies">
    <documentation>
      xmlns="http://schemas.xmlsoap.org/wsdl/"
    </documentation>
    <wsdl:input message="tns:GetTheatersAndMoviesSoapIn"/>
    <wsdl:output message="tns:GetTheatersAndMoviesSoapOut"/>
  </wsdl:operation>
</wsdl:portType>

<wsdl:binding name="MovieInformationSoap" type="tns:MovieInformationSoap">
  <soap:binding style="document"
                transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="GetTheatersAndMovies">
    <soap:operation soapAction="http://www.ignite.com/whatsshowing/GetTheate
rsAndMovies" style="document"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>

<wsdl:service name="MovieInformation">
  <wsdl:port binding="tns:MovieInformationSoap"
            name="MovieInformationSoap">
    <soap:address location="http://www.ignite.com/webservices/ignite.whatssh
owing.webservice/moviefunctions.asmx"/>
  </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```

Question 4 [Cont'd]

You may find the following codes useful:

```
<?php
require_once('nusoap/nusoap.php');
$key = $_POST['txt_key'];

$wsdl = "http://ghettodriveby.com/soap/?wsdl";
$client = new nusoap_client($wsdl, 'wsdl');
//input requires only one parameter, called word, type string
$params = array('word'=>$key);
$result = $client->call('getRandomGoogleSearch', $params);
    if ($client->fault) {
        echo '<p><b>Fault: ' ;
        print_r($result);
        echo '</b></p>';
    } else {
        // Check for errors
        $err = $client->getError();
        if ($err) {
            // Display the error
            echo '<p><b>Error: ' . $err . '</b></p>';
        } else {
            // Display the result

            print_r($result); //dump the contents of the result (added here
only to illustrate the output

            //output consists of 2 elements, one of which is called image
            $img_src=$result['image'];
            echo "<img src= '$img_src' />";
        }
    }
?>
```

END OF QUESTION PAPER

/ph