**Lab 4 - RSS Search**

**NOTE: YOU WILL NEED INTERNET CONNECTION FOR THIS EXERCISE. IF THE LAB DOES NOT HAVE NET CONNECTION, THEN USE THE FIVE (5) XML FILES ATTACHED ALONG WITH THIS DOCUMENT (you will have to demo the code with a fixed set of search strings and assume that each one returns one of the XMLs as the match. Right now if NET connection is not available, you can only demo the code with the following search strings: “Sachin”, “Einstein”, “Apple”, “Personal Development” and “AJAX”)**

**Instructions:** PLEASE READ CAREFULLY

1. Build an RSS-based search.

2. The main page has a text box (for searching) and also a button labeled "Search".

3. When the user enters a piece of text and clicks the "Search" button, an AJAX call needs to be made to your server(localhost) which performs a search on the web. (This is because we cannot afford our own Search engine).

4. The search needs to be RSS based because if the search returns an HTML (like how normal Google search does), then the server will have a lot of work to do to parse the html and extract meaningful parts. Hence an RSS (which is actually an XML) needs to be returned. To do this, use bing's RSS-based search engine.

**The URL is "http://www.bing.com/search?q=sachin&format=rss";** (Here "sachin" is the word you are searching for).

So first extract $\_GET in the PHP file and pass the variable into the URL. The return value is an xml string. Send this back to the client.

The xml string looks like this:

<?xml version="1.0"?>

<rss version="2.0">

<channel>

<title>...</title>

<description>...</description>

<item>

<title>...</title>

<link>...</link>

...

</item>

.....

</channel>

</rss>

5. Now build a <div> based display ( similar to the auto-suggest we tried in class)

Hints:

1. You can have 2 divs one an outer and the other an inner (there many inner divs)

2. You have to make the AJAX call using plain XHR. Get the returned xml in xhr.responseXML. This is possible if you use header("Content-type:text/xml") on the server side. You can then navigate the XML manually using the root node (brute force method from UNIT1. To get the root node, use the code – “root = xhr.responseXML.documentElement”). You need to extract the <link> and <title> nodes for each <item> node. Each <item> node represents ONE search result. So if the query returned 20 results, the XML will have 20 <item> nodes. For each <item> node the “href” (url to the actual search result page) will be in the <link> node and the description (title) which will be shown as innerHTML of the anchors, will be in the <title> node. You can use “root.getElementsByTagName(‘item’)” to get ALL item nodes. Here “root” refers to the root element of the XML DOM and is NOT a keyword.

4. Once you have the values, create individual divs using document.createElement("div").

Add the link and the title values extracted in the previous step, into the divs

5. Append the divs to the outer div and then position the outer div properly.

6. On the server side, use:

$retstr = file\_get\_contents(url); (See point.4 above in the ‘instructions’ part).

The url is bing's url and $retstr will have the RSS feed when the call is successful.

**If NET connection is not available, then follow the instructions given at the top of this document.**

See the attached images (below) to get some idea.

NOTE: You may get several results. Choose a reasonable number and display only so many search results. Each search result is an anchor (link) to the actual site. You can see 4 such links in the image below.

**Before search:**



**After Search:**

