NAICS Association Business Intelligence API

Version 3.0.1.1

Updated 08/12/2020

Table of Contents

0.0 Test the BizAPI without coding

You can test the API service without doing any coding. Just go to this link, https://www.naics.com/naics-api-test/, enter your credentials and fill the form with the Company record data. When submitted, you will see the JSON result of your request.

1.0 Making a Request

The API is organized around REST. All requests should be made over SSL. All request and response bodies, including errors, will be encoded in JSON. Requests are Authenticated via Basic Authentication. A maximum of 3 Requests can be made per rolling second. If you need this temporarily boosted for a large project, please reach out to your API Account contact or APICloudSolutions@NAICS.com for support.

The BizAPI Utilizes Basic Authentication. Credentials are given to you during your Account Activation.

For Example:

Username: JohnDoeInc

Password: xxxx xxxx xxxx xxxx xxxx xxxx

Spaces in Password MUST be kept

1.1 POST

POST https://www.naics.com/wp-json/naicsapi/v1/cosearch

The POST body should contain the following fields (replace Fields in Yellow with Company Data):

```
{
    "companyName": "Apple Inc",
    "address": "1 Apple Park Way",
    "city": "Cupertino",
    "state": "CA",
    "postalCode": "55144",
    "country": "US",
    "phone": "4089961010"
}
```

See Sample Code in ASP, PHP, and JavaScript with jQuery at the bottom of this Document.

1.2 Response

The BizAPI returns standard Success and Error Codes.

HTTP Status codes

200 OK
400 Syntax Error
401 Credentials Missing or Invalid.
403 "Request not submitted due to lack of searches. Contact

apicloudsolutions@naics.com or call 973-625- 5626 to purchase more searches. "

429 "Too many requests. Please limit your reqests to 3 per second"

500 Internal Server Error

If submission successfully processes (code 200) but no match can be found, the following JSON will be returned. `Text in Search Terms will reflect Company information submitted`


```
{
    "Search Terms": {
        "Company Name": "Not a Real Company",
        "Street Address": " 123 Bad Street Address",
        "City": "Wrong City",
        "State": "Wrong State",
        "Zip Code": "33556",
        "Country": "US",
        "Phone": ""
    },
    "Matching Data": {
        "DUNS #": "",
        "Matches Remaining": 94935
    },
    "Appended Data": {
        "Location Type": "",
        "8 Digit SIC 1": "",
        "8 Digit SIC 2": "",
        "NAICS 1 Description": "No match found"
    }
}
```


If a match can be found, the following JSON will be returned, based on the record layout Established when Credentials were created. See Record Layouts for more information. Below is an example of all possible fields (results continue onto next page).


```
{
    "Search Terms": {
        "Company Name": "Apple Inc",
        "Street Address": "1 Apple Park Way",
        "City": "Cupertino",
        "State": "CA",
        "Zip Code": "55144",
        "Country": "US",
        "Phone": "4089961010"
    },
    "Matching Data": {
        "BEMFAB": "M",
        "Match Grade": "AAAAAZA",
        "Confidence Code": "10",
        "DUNS #": "06-070-4780",
        "Matches Remaining": 94935
    },
    "Appended Data": {
```

```
        "Company Name": "Apple Inc.",
        "Secondary Business Name": "Apple",
        "Address Source": "physical address",
        "Street Address": "1 Apple Park Way",
        "PO Box": "",
        "State/Province": "CA",
        "City": "Cupertino",
        "ZIP Code": "95014-0642",
        "Country": "USA",
        "Phone": "4089961010",
        "URL": "www.apple.com",
        "CEO Title": "Chief Executive Officer",
        "CEO Name": "Timothy D Cook",
        "CEO First Name": "Timothy",
        "CEO Last Name": "Cook",
        "Line of Business": "Radio and TV communications equipment,nsk",
        "Location Type": "Headquarters",
        "Year Started": "1977",
        "Employees on Site": "2000",
        "Employees Total": "137000",
        "Sales Volume": "260,174,000,000",
        "4 Digit SIC 1": "3663",
        "4 Digit SIC 1 Description": "Radio and T.v. Communications Equipment",
        "4 Digit SIC 2": "3571",
        "4 Digit SIC 2 Description": "Electronic Computers",
        "8 Digit SIC 1": "36639906",
        "8 Digit SIC 1 Description": "Mobile communication equipment",
        "8 Digit SIC 2": "35719904",
        "8 Digit SIC 2 Description": "Personal computers (microcomputers)",
        "NAICS 1 Code": "334220",
        "NAICS 1 Description": "Radio and Television Broadcasting and Wireless Communi
cations Equipment Manufacturing",
        "NAICS 2 Code": "334111",
        "NAICS 2 Description": "Electronic Computer Manufacturing",
        "Subsidiary Indicator": "not a subsidiary site",
        "Global Ult DUNS #": "060704780",
        "Global Ult Bus. Name": "Apple Inc.",
        "Global Ult Indicator": "Y",
        "Global Ult State/Province": "California",
        "Global Ult Country": "USA",
        "Domestic Ult DUNS #": "060704780",
        "Domestic Ult Bus. Name": "Apple Inc.",
        "Domestic Ult State/Province": "California",
        "Domestic Ult Country": "USA",
        "HQ/Parent Ult DUNS #": "",
        "HQ/Parent Ult Bus. Name": "",
        "HQ/Parent State/Province": "",
        "HQ/Parent Country": "",
        "Hierarchy Code": "1",
        "# of Family Members": "727"
    }
}
```

1.3 Record Layout Types

Each Record Layout utilizes a unique price tier. Contact APICloudSolutions@NAICS.com or your NAICS Representative to acquire Pricing.

Each Successful Response will include the *Search Terms (Input Data)*, *Matching Data*, and *Appended Data*. The *Appended Data* Returned will be based on one of the following Layouts:

NAICS & SIC Layout

Includes the Following:

- 4 Digit SIC 1
- 4 Digit SIC 1 Description
- 4 Digit SIC 2
- 4 Digit SIC 2 Description
- 8 Digit SIC 1
- 8 Digit SIC 1 Description
- 8 Digit SIC 2
- 8 Digit SIC 2 Description
- NAICS 1 Code
- NAICS 1 Description
- NAICS 2 Code
- NAICS 2 Description

Telemarketing Layout

Includes *NAICS & SIC* content PLUS:

- Company Name
- Secondary Business Name
- Address Source
- Street Address
- PO Box
- State/Province
- City
- ZIP Code
- Phone
- CEO Title
- CEO Name
- CEO First Name
- CEO Last Name
- Line of Business
- Location Type
- Year Started

Enhanced Telemarketing with Employee Figures Layout

Includes *NAICS & SIC* + *Telemarketing* content PLUS:

- Employees on Site
- Employees Total

Enhanced Telemarketing with Sales Figures Layout

Includes *NAICS & SIC* + *Telemarketing* content PLUS:

- Sales Volume

Prospecting Layout

Includes *NAICS & SIC* + *Telemarketing* content PLUS:

- Employees on Site
- Employees Total
- Sales Volume

Prospecting with Linkage Layout

Includes *NAICS & SIC* + *Telemarketing* + *Prospecting* content PLUS:

Fields Returned	Explanation of Fields
Subsidiary Indicator	Indicates if Appended Record is/is not 50%+ own by a Parent.
Global Ult Indicator	Indicates if Appended Record is/is not the Global Ult Record.
Global Ult DUNS # Global Ult Bus. Name Global Ult State/Province Global Ult Country	The Global Ultimate Company is the highest family member in any country within the Appended business record's Family Tree. A subject may be its own Global Ultimate.
Domestic Ult DUNS # Domestic Ult Bus. Name Domestic Ult State/Province Domestic Ult Country	The Domestic Ultimate Company is the highest family member in the same country as the Appended business record. A subject may be its own Domestic Ultimate.
HQ/Parent Ult DUNS # HQ/Parent Ult Bus. Name HQ/Parent State/Province HQ/Parent Country	If Appended Record is a Headquarters or Single Location, then these fields reflect the Parent Company which has 50%+ ownership of the Appended business record. If the Appended Record is a Branch Location, then these fields reflect the Headquarter Location the Branch reports to.
Hierarchy Code	The hierarchy code is a two-digit field which determines the records relative position in a family tree by indicating its relationship to other records. The hierarchy code functions in the following way- Global Ultimates have a hierarchy code of- 01. Subsidiaries have a hierarchy code of one greater than their parents. Branches have a hierarchy codeequal to their headquarters.
# of Family Members	The total number of family members within a family tree. This countincludes the global ultimate itself, all global ultimates' branch locations, all subsidiaries, and all subsidiary branch locations.

1.4 Improving Input Data Quality

The User should get data for their match project from the highest and best source of data they have internally. (Often, highest ≠ best)
• By Highest, we mean the source that is closest to the point of data entry (when they open a new account, the data is entered at that point.)
• By Best, we mean the source that has the most accurate data contained in the match input fields and it should have the most complete data available.

Other factors, which will affect input file accuracy, include situations where content is contained in the wrong field. For instance, we frequently see situations where the User will contain information in their Address field, which in fact is not the customer's address.

Some common examples of this issue:

1. Where the address field contains information other than an address. For instance:
   a) A person's name or
   b) ATTN TO:
   c) A Department Name
   d) A second Company Name

2. Where the address correlates to a Different Business then provided in the record. This frequently occurs where the User sells through dealers and/or Value Added Resellers (VARs). It can also occur if the Business on Record provides the Address of the Third Party they representing instead of their own.

Where possible, the User should eliminate or correct the bias these inaccuracies cause before submitting a request.

Request Parameters

The BizAPI utilizes intuitive Matching software to match against Records written in many different styles.

Below are some insights to ensure you are maximizing the opportunities to Append successfully:

- Each Field can handle over 200 characters. 65 characters or less is generally better for matching.
- No layout needs to be specified when submitting a request. Layout is established upon API Credential Activation.
- Special Characters such Ò ‹ œ and © will not be acknowledged by the API and may be turned into a question mark (?) if a match is found. Avoid using special characters if you can.
- "companyName": "<company name>", should contain only One company name. If working with more than One name for the same organization, consider running a second round using the Additional Name if the first submission does not append.
- "*address*": "<*address*>" Should only Contain the Street Address OR a PO Box #.
- "*phone*": "<*phone*>" can handle most Standard Phone Formats. Don't include Extensions. Use a string of numbers with no Special characters to ensure the highest chance for a match.
- If a submission has a Zip code but lacks City and/or State information, our system will use the Primary Postal City and State linked to the Zip code in order to allow for an append to be possible.
- Use Data written in a Standard Postal Code convention when able.
  Ex. Avenue can be written as: Av, Ave, Aven, Avenu, Avenue, Avn, Avnue.

Completeness

The customer should supply as many of the fields in the input layout as possible. Not only should they supply as many of the fields, but also the fields should be populated as completely as possible. Only Records matched at a Confidence score of 7 or higher will be appended, so Records that lack 3 or more fields are not likely to match at a sufficient Confidence Level to Append.

If you have a significantly number of Business Records that lack Complete address information, consider utilizing our Batch Append Service to enrich your data based on looser Parameters. Reach out to your NAICS Representative or Appends@naics.com to discuss your data challenges.

1.5 Test Endpoint (Sandbox) for the BizAPI Service

A Test Endpoint has been created to allow Users to submit fake responses for testing with Third Party Applications without having to consume existing credits.

POST https://www.naics.com/wp-json/naicsapi/v1/cosearchtest

The Sandbox requires basic authentication just like the live endpoint.

To Yield a Successful append result, the POST body should contain the following fields (no need to replace Fields in Yellow with Real Company Data). With one exception (explained below), any request will reflect the Same Layout Established for the Account, but with Fake Business Data.

```
{
    "companyName": "Test Company ABC",
    "address": "123 Test Ave ",
    "city": " New York ",
    "state": "NY",
    "postalCode": "55555",
    "country": "USA",
    "phone": "5555555555"
}
```

To yield an Unsuccessful append result, the POST body must contain the exact following fields. The Unsuccessful Response will be in the same format as in the Live Endpoint.

```
{
    "companyName": "Bad Company ABC",
    "address": "123 Test Ave ",
    "city": "New York",
    "state": "NY",
    "postalCode": "55555",
    "country": "USA",
    "phone": "5555555555"
}
```

2.0 Understanding Matching Data

Matching Data is included to help you understand the Quality of the Appended data Matched to your Input Data.

There are 5 Fields included:

"BEMFAB":
"Match Grade":
"Confidence Code":
"DUNS #":
"Matches Remaining":

2.1 Bemfab

The Bemfab indicator gives insight about the Marketability of the Appended data. It will return a Single Letter Response or return as blank. These are the meanings behind each Possible Return Value:

Blank: Marketability is undefined.

M = Matched to Marketable Record File.

A = Dun's numbered record that has been flagged as undeliverable. Industry will be appended.

D = At the customer's request the record has been De-listed. Can't sell company name but can append Industry code.

O = Unconfirmed record or confirmed as Out of business.

2.2 Match Grade

The Match Grade will return a 7 Letter String when a Successful Match occurs. Each letter indicates how successfully a piece of the Input Data was matched to the Database Record.

Here is an example of how the String might look: ABAAAFZ

This is the sequence in which the Match Grade string reflects each data element.

1. Company Name
2. Street Address Number
3. Street Name
4. City
5. State
6. PO Box*
7. Telephone

Letter Grades should be interpreted in the following way:

A - Same: ABC WIDGET MFG vs. ABC Widgeting MFG

B - Similar: ABC Widget MFG vs. ABC MFG

F - Different: ABC Widget MFG vs. XYZ MFG

Z - Null: One or Both are Null (Blank)

*Note: When you submit a record, you can supply Either a Street Address or a PO Box in the "*address*": "<*address*>" field. If the Record fails to Match on Street Address, it is recommended you try again using the PO Box Address.

2.3 Confidence Code

Confidence codes indicate the strength of match or our “Confidence Level” in that particular match. 10 is the highest confidence level for a matched record down to 4 which is the loosest match. The API only appends Records with a Confidence Level of 7 or Higher. Commonly all records matched at 7 or higher are strong matches. 6’s can be strong matches, but not consistently enough to allow through the API. 5’s and 4’s are either just a Company Name or Physical Address match so we typically don’t append to them.

2.4 DUNS

The D&B D-U-N-S® Number is a unique nine-digit identifier for businesses. It is used to establish a business credit file, which is often referenced by lenders and potential business partners to help predict the reliability and/or financial stability of the company in question. D-U-N-S, which stands for data universal number system, is used to track and maintain accurate and timely information on +265M global businesses. (source- http://www.dnb.com/duns-number.html)

2.5 Matches Remaining

This Field indicates how Many Search Credits you have remaining before the API will stop working. To Increase your Credits, or request assistance please email us at APICloudSolutions@NAICS.com, or call 973-625-5626.

Go to Next Page for Code Snippets

CODE SNIPPET FOR PHP

```php
<?php
	$username = "xxx"; //Replace xxx with your username.
	$password = "xxx"; //Replace xxx with yor password.

	$credentials = base64_encode($username.":".$password);

	if (!empty($_POST)){
		// Collect data posted from form into an array.
		$companyInfo = array(
			'companyName' => $_POST['companyName'],
			'address' => $_POST['address'],
			'city' => $_POST['city'],
			'state' => $_POST['state'],
			'country' => $_POST['country'],
			'phone' => $_POST['phone'],
			'postalCode' => $_POST['postalCode'],
		);

		// Prepare REST request
		// Get cURL resource
		$ch = curl_init();

		// Set url
		curl_setopt($ch, CURLOPT_URL, 'https://www.naics.com/wp-json/naicsapi/v1/cosearch');
		// Set method
		curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'POST');
		// Set options
		curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
		// Set headers
		curl_setopt($ch, CURLOPT_HTTPHEADER, [
		 "Authorization: Basic ".$credentials,
		 "Content-Type: application/json; charset=utf-8",
		]
		);

		// Create body
		$body = json_encode($companyInfo);

		// Set body
		curl_setopt($ch, CURLOPT_POST, 1);
		curl_setopt($ch, CURLOPT_POSTFIELDS, $body);

		// Send the request & save response to $response
		$response = curl_exec($ch);

		if(!$response) {
		 die('Error: "' . curl_error($ch) . '" - Code: ' . curl_errno($ch));
		} else {
		 $status = "Response HTTP Status Code : " . curl_getinfo($ch, CURLINFO_HTTP_CODE);
		 // Convert JSON reply to a PHP array
```

```
                $respArray = json_decode($response, true);
                // Build HTML output
                $htmlContent = "<ul>";
                foreach ($respArray as $section=>$sectionArray){
                        $htmlContent .= "<li><h4>";
                        $htmlContent .= $section;
                        $htmlContent .= "</h4>";
                        $htmlContent .= "<ul>";
                        foreach ($respArray[$section] as $key=>$value){
                                $htmlContent .= "<li>";
                                $htmlContent .= $key.": <strong>".$value."</strong>";
                                $htmlContent .= "</li>";
                        }
                        $htmlContent .= "</ul></li>";
                }
                $htmlContent .= "</ul>";
                }
                // Close request to clear up some resources
                curl_close($ch);
        }
?>
<!DOCTYPE html>
<html lang="en">
        <head>
                <meta charset="UTF-8">
                <title>NAICS Association API Sample Code</title>
        </head>
        <body>
                <h1>NAICS Association API Sample Page</h1>
                <form method="post">
                        Company name*:<br>
                        <input type="text" name="companyName" required><br>
                        Address:<br>
                        <input type='text' name='address'><br>
                        City:<br>
                        <input type='text' name='city'><br>
                        State*:<br>
                        <input type='text' name='state' required><br>
                        Country*:<br>
                        <input type='text' name='country' required><br>
                        Phone:<br>
                        <input type='text' name='phone'><br>
                        Postal Code:<br>
                        <input type='text' name='postalCode'><br>
                        <input type='submit' value='Submit'>
                </form>
                <p><?php echo $status; ?></p>
                <?php echo $htmlContent; ?>
        </body>
</html>
```

CODE SNIPPET FOR ASP

```
<%
if Request.Form("go")=1 then
 username=Request.Form("User")
 password=Request.Form("pwd")

end if
%>
<!DOCTYPE html>
<html lang="en">
        <head>
                <meta charset="UTF-8">
                <title>NAICS Association API Sample Code</title>
                <script src="https://code.jquery.com/jquery-3.2.1.min.js" integrity="sha256-
hwg4gsxgFZhOsEEamdOYGBf13FyQuiTwlAQgxVSNgt4=" crossorigin="anonymous"></script>
                <script type="text/javascript">
                        $(document).ready(function(){
                                $("#searchForm").submit(function(e){
                                        e.preventDefault();

                                        // The username and password below will be saved in clear text and available
to anybody who has access to the source code of this page.
                                        // This sample code is for testing and demonstration purposes only. Passwords
and usernames should never be stored this way on
                                        // a page that can be accessed in any way by users who are not trusted.
                                        var username = "<%=username%>"; //replace with your Username
                                        var password = "<%=password%>"; //replace with your password
                                        var form = $(this);
                                        searchFormData = form.serializeArray();
                                        var htmlContent = "";
                                        // send ajax
                                        jQuery.ajax({
                                          url: "https://www.naics.com/wp-json/naicsapi/v1/cosearch",
                                          type: "POST",
                                          headers: {
                                            "Authorization": "Basic " + btoa(username + ":" + password),
                                            "Content-Type": "application/json; charset=utf-8",
                                          },
                                          contentType: "application/json",
                                          data: JSON.stringify(convertFormData(searchFormData))
                                        })
                                        .done(function(naicsResponse, textStatus, jqXHR) {
                                          $( "#requestStatus" ).html("HTTP Request Succeeded: " + jqXHR.status);
                                                $.each(naicsResponse, function(section, items){
                                                        htmlContent += "<li><h4>";
                                                        htmlContent += section;
                                                        htmlContent += "</h4>";
                                                        htmlContent += "<ul>";
                                                        $.each (items, function(key, item){
```

```
                                                htmlContent += "<li>";
                                                htmlContent += key + ": <strong>" + item +
"</strong>";
                                                htmlContent += "</li>";
                                        });
                                 htmlContent += "</ul></li>";
                                });
                                $( "#responseList" ).html(htmlContent);
                        })
                        .fail(function(jqXHR, textStatus, errorThrown) {
                                $( "#requestStatus" ).html("HTTP Request Failed: " + jqXHR.status);
                                var failResponse = JSON.parse(jqXHR.responseText);
                                $.each(failResponse, function(key, item){
                                        if (key != "data"){
                                                htmlContent += "<li>";
                                                htmlContent += key + ": <strong>" + item +
"</strong>";
                                                htmlContent += "</li>";
                                        }
                                });
                                $( "#responseList" ).html(htmlContent);
                        })
                        .always(function() {
                                // Clear form entries and reset default values after everything else
has completed.
                                  $('#searchForm').trigger("reset");
                        });
                });
        });
        // Put form data into array that can converted into JSON
        function convertFormData(data) {
                var unindexed_array = data;
                var indexed_array = {};
                $.map(unindexed_array, function(n, i) {
                        indexed_array[n['name']] = n['value'];
                });
                return indexed_array;
        }
    </script>
  </head>
  <body>
    <h1>NAICS Association API Sample Page</h1>
<%
 if Request.Form("go")<>"1" then
  Response.Write("<form name=Credentials action=""ajaxAPI.asp"" method=post><input type=hidden name=go value=""1"">")
  Response.Write("<table><tr><td align=right>User:</td><td><input type=password name=User size=50></td></tr>")
  Response.Write("<table><tr><td align=right>Password:</td><td><input type=password name=pwd size=50></td></tr>")
  Response.Write("<table><tr><td align=right></td><td><input type=submit value=""Go""></td></tr></table></form>")
 else
%>
    <form id="searchForm" method="post">
```

```
                    Company name*:<br>
                    <input type="text" name="companyName" required><br>
                    Address:<br>
                    <input type='text' name='address'><br>
                    City:<br>
                    <input type='text' name='city'><br>
                    State*:<br>
                    <input type='text' name='state' required><br>
                    Country*:<br>
                    <input type='text' name='country' required><br>
                    Phone:<br>
                    <input type='text' name='phone'><br>
                    Postal Code:<br>
                    <input type='text' name='postalCode'><br>
                    <input id="submitBtn" type='submit' value='Submit'>
                </form>
<%
 end if
%>
                <p id="requestStatus"></p>
                <ul id="responseList"></ul>
        </body>
</html>
```

CODE SNIPPET FOR JavaScript with jQuery

```
<!DOCTYPE html>
<html lang="en">
	<head>
		<meta charset="UTF-8">
		<title>NAICS Association API Sample Code</title>
		<script src="https://code.jquery.com/jquery-3.2.1.min.js" integrity="sha256-
hwg4gsxgFZhOsEEamdOYGBf13FyQuiTwlAQgxVSNgt4=" crossorigin="anonymous"></script>
		<script type="text/javascript">
			$(document).ready(function(){
				$("#searchForm").submit(function(e){
					e.preventDefault();

					// The username and password below will be saved in clear text and available
ot anybody who has access to the source code of this page.
					// This sample code is for testing and demonstration purposes only. Passwords
and usernames should never be stored this way on
					// a page that can be accessed in any way by users who are not trusted.
					var username = "xxx"; // Replace xxx with your username.
					var password = "xxx"; // Replace xxx with yor password.
					var form = $(this);
					searchFormData = form.serializeArray();
					var htmlContent = "";
					// send ajax
					jQuery.ajax({
					  url: "https://www.naics.com/wp-json/naicsapi/v1/cosearch",
					  type: "POST",
					  headers: {
					    "Authorization": "Basic " + btoa(username + ":" + password),
					    "Content-Type": "application/json; charset=utf-8",
					  },
					  contentType: "application/json",
					  data: JSON.stringify(convertFormData(searchFormData))
					})
					.done(function(naicsResponse, textStatus, jqXHR) {
					  $( "#requestStatus" ).html("HTTP Request Succeeded: " + jqXHR.status);
						$.each(naicsResponse, function(section, items){
							htmlContent += "<li><h4>";
							htmlContent += section;
							htmlContent += "</h4>";
							htmlContent += "<ul>";
							$.each (items, function(key, item){
								htmlContent += "<li>";
								htmlContent += key + ": <strong>" + item +
"</strong>";
								htmlContent += "</li>";
							});
						 htmlContent += "</ul></li>";
						});
						$( "#responseList" ).html(htmlContent);
					})
					.fail(function(jqXHR, textStatus, errorThrown) {
						$( "#requestStatus" ).html("HTTP Request Failed: " + jqXHR.status);
						var failResponse = JSON.parse(jqXHR.responseText);
```

```
                                        $.each(failResponse, function(key, item){
                                                if (key != "data"){
                                                        htmlContent += "<li>";
                                                        htmlContent += key + ": <strong>" + item +
"</strong>";
                                                        htmlContent += "</li>";
                                                }
                                        });
                                        $( "#responseList" ).html(htmlContent);
                                })
                                .always(function() {
                                        // Clear form entries and reset default values after everything else
has completed.
                                   $('#searchForm').trigger("reset");
                                });
                        });
                });
                // Put form data into array that can converted into JSON
                function convertFormData(data) {
                        var unindexed_array = data;
                        var indexed_array = {};
                        $.map(unindexed_array, function(n, i) {
                                indexed_array[n['name']] = n['value'];
                        });
                        return indexed_array;
                }
        </script>
    </head>
    <body>
        <h1>NAICS Association API Sample Page</h1>
        <form id="searchForm" method="post">
                Company name*:<br>
                <input type="text" name="companyName" required><br>
                Address:<br>
                <input type='text' name='address'><br>
                City:<br>
                <input type='text' name='city'><br>
                State*:<br>
                <input type='text' name='state' required><br>
                Country*:<br>
                <input type='text' name='country' required><br>
                Phone:<br>
                <input type='text' name='phone'><br>
                Postal Code:<br>
                <input type='text' name='postalCode'><br>
                <input id="submitBtn" type='submit' value='Submit'>
        </form>
        <p id="requestStatus"></p>
        <ul id="responseList"></ul>
    </body>
</html>
```

3.0 Data Flow Architecture

The following Architecture demonstrates the flow of Data from start to finish of the Request. Input data is not stored on any server involved, at any point during the process.