IDOLOGY

ExpectID® IQ

8.43 8. 2014

API Integration Guide

Table of Contents

Introduction 3
Definitions 3
API Overview 3
STEP 1 – Building the Request 5
STEP 2 – Sending the Request 7
STEP 3 – Processing the XML Response from ExpectID 8
Sample XML Responses 9
Optional Differentiating Question 11
Critical Errors........ 13
STEP 4 – Processing the XML Response from an ExpectID IQ transaction 14
STEP 5 – Submitting the Answers 18
The Answers Response 19
PHP Sample Code for Answer Submission and Response 23
Other XML Output 24
Velocity 24
ExpectID Score 28
ID Notes 28
STEP 6 – Input Considerations 33
Contacting IDology Customer Service 35

Introduction

ExpectID® is an identity verification service that is used to verify if an identity is real without requiring consumer interaction. ExpectID relies on IDology's analytics to assess the components of an ID and return a result.

ExpectID® IQ is a higher level of verification that enhances the basic ExpectID® identity verification process by allowing you to ask questions from consumers, submit the answers back to IDology and receive a result on how many questions were answered correctly. The data used to generate the questions is broadly based on information found in thousands of trusted data sources rather than payment or credit card information.

This guide is intended for IT teams responsible for integrating IDology's services into an existing web site or application.

Definitions

- **Client:** The party integrating its application with the IDology Web service.
- **Subject:** The person or set of credentials that is the subject of the authentication process.
- **End user:** An individual using IDology's verification services to verify a subject's identity.
- **IDCenter:** IDology's web portal which is used to process verifications and serves as an on-demand change management tool for the Client.
- **Enterprise:** The Client's account, settings, and business rules established and maintained through the IDCenter.
- **Configuration:** The process of defining the Client's Enterprise business rules and settings in the IDCenter.
- **Integration:** The process of implementing IDology's services, via the IDology API, within an existing application or web site.
- **Touch Points:** The point in the business process where ExpectID services are integrated. This business decision needs to be made in conjunction with the Enterprise Configuration, prior to beginning the integration of ExpectID services into the client's application. The most common touch points for integration are point of entry, account origination, point of sale, and account modifications.

Overview

Integrating both ExpectID and ExpectID IQ involves three basic steps:

1. Making a secure connection to the ExpectID service
2. Submitting a request
3. Receiving a formatted XML response.

This guide includes sample PHP code, but other code examples are available upon request.

ExpectID and ExpectID IQ are both provided as a REST-based web service accessed via HTTPS POST. Requests are submitted as name and value pairs in the body of an HTTPS POST and responses are returned as XML messages *(refer to the IDology Integration, XML guide for more information about XML responses.)*

Before you begin, make certain that you have permissions on the server to establish outgoing SSL (HTTPS) connections and any necessary additional libraries for your installation (CURL, etc). IDology also restricts access to your Enterprise account by IP address, so make sure IDology has included your infrastructure's IP address(es) in the IP whitelist for your company. Please note that you should only access our services via the fully qualified domain name (FQDN). We may at any time direct traffic to our secondary location by changing our IP addresses in DNS. If you are not using the FQDN, you may not be able to reach our services at all times. Finally, you need your IDology username and password for requests going through the API. For security purposes, the password you set up for your API account needs to be 12-40 characters.

Also, make certain the Enterprise Configuration is complete in the IDCenter. The configuration of thresholds, rules, and messaging in the IDCenter are business decisions that reflect business policies and objectives of the Enterprise and therefore should not be altered greatly during the integration.

There are additional steps associated with a complete ExpectID IQ transaction:

1. **ExpectID IQ Response.** If the subject is eligible for questions, and the business rules that specify questions for the Enterprise are triggered, questions are returned through the API so that they can be presented. If questions cannot be generated for the subject, an error message will be returned.

2. **ExpectID IQ Answer submission.** Once the subject answers the questions, submit the answers to IDology. The service will respond with the number of questions that were answered correctly and the customizable response message the client has established for that result.

Step 1 – Building the Request

Begin by creating an array of values being submitted (required fields in red):

```
// ID FIELDS - PLEASE INCLUDE ALL FIELDS (EVEN IF BLANK)
$fields = array(
        'username'             => '',  // YOUR ExpectID USERNAME (16)
        'password'             => '',  // YOUR ExpectID PASSWORD
        'invoice'              => '',  // YOUR INVOICE OR ORDER NUMBER (30)
        'amount'               => '',  // ORDER AMOUNT
        'shipping'             => '',  // SHIPPING AMOUNT
        'tax'                  => '',  // TAX AMOUNT
        'total'                => '',  // TOTAL AMOUNT (SUM OF THE ABOVE)
        'idType'               => '',  // TYPE OF ID PROVIDED
        'idIssuer'             => '',  // ISSUING AGENCY OF ID
        'idNumber'             => '',  // NUMBER ON ID
        'paymentMethod'        => '',  // PAYMENT METHOD
        'firstName'            => '',  // FIRST NAME
        'lastName'             => '',  // LAST NAME
        'address'              => '',  // STREET ADDRESS
        'city'                 => '',  // CITY
        'state'                => '',  // STATE (2)
        'zip'                  => '',  // 5-DIGIT ZIP CODE (5)
        'ssnLast4'             => '',  // LAST 4 DIGITS OF SSN (4)
        ‘ssn’                  => “,  // FULL SSN
        'dobMonth'             => '',  // MONTH OF BIRTH (2)
        ‘dobDay’               => “,  // DAY OF BIRTH (2)
        'dobYear'              => '',  // YEAR OF BIRTH (4)
        'ipAddress'            => '',  // IP ADDRESS
        ‘emailAddress’         => “,  // EMAIL ADDRESS
        ‘telephone’            => “,  // PHONE NUMBER
        'sku'                  => '',  // SKU
        'uid'                  => '',  // USER ID (EXTERNAL APPLICATION)
        ‘altAddress’           => '',  // ALTERNATE STREET ADDRESS
        ‘altCity’              => “,  // ALTERNATE CITY
        ‘altState’             => “,  // ALTERNATE STATE (2)
        ‘altZip’               => “,  // ALTERNATE 5-DIGIT ZIP CODE (5)
);
```

NOTES:

- Many of these fields are optional. The minimum required for a successful transaction are username, password, firstName, lastName, address, and zip. Results improve with addition of ssnLast4 and/or dob information. YOB is the minimum DOB information IDology accepts. For optimal performance and full change management functionality, we recommend submitting all the fields above even if the value is blank.
- dobMonth, dobDay, dobYear, ssn, and ssnLast4 may or may not be not be required fields, depending on the Enterprise configuration.
- You must include ZIP code AND/OR City + State.
- The Alternate Address fields can be used in the IDology API to submit a secondary address for verification, like a shipping address.
- Fields are generally limited to 40 characters in length, except as noted above (Character length limits are in parenthesis after field description).

- If submitting an IP Address, please include periods in the address, for example – 11.111.111.11
- While most of these values will be populated by the contents of a form on your website, it is very important that the username and password fields be hard coded to prevent possible user tampering. These fields should ***never*** be included in the HTML source of a public web page.

STEP 2 – Sending the Request

Once the field values are set, construct the XML query and establish a connection to the server at this URL:

https://web.idologylive.com/api/idiq.svc

```
// SET IDOLOGY SERVICE URL
$url = 'https://web.idologylive.com/api/idiq.svc';

// GENERATE QUERY FROM FIELDS ARRAY
$query = http_build_query($fields);

// DEFINE CONNECTION OPTIONS
$options = array(
    'http'   => array(
            'method' => "POST",

            'header' => "Host: web.idologylive.com\r\n".
                        "Content-type: application/x-www-form-urlencoded\r\n".
                        "User-Agent: Mozilla 4.0\r\n".
                        "Content-length: ".strlen($query)."\r\n".
                        "Connection: close\r\n\r\n",

            'content' => $query,
    )
);
```

STEP 3 – Processing the XML Response from ExpectID

So far you have built our array of values, submitted the request, and read the resulting output into a new variable called $content. The final step is to read the resulting output. Because individual implementations may use the results in different ways, the following example shows how fields can be retrieved using either regular expression pattern-matching or PHP's built-in XML parser.

```
// CREATE STREAM CONTEXT
$context = stream_context_create($options);

// CONNECT TO IDOLOGY URL AND RETRIEVE RESPONSE
$result = file_get_contents($url, false, $context, -1, 40000);

// CREATE ARRAY TO STORE RESULTS
$results = array();

// PARSE XML RESPONSE
$dom = new DomDocument();
$dom->loadXML($content);

// CONVERT DOM OBJECT INTO ARRAY
foreach ($dom->documentElement->childNodes as $item) {
    if ($item->nodeType == 1) {
        $name = $item->nodeName;
        $value = $item->textContent;

        if ($item->childNodes->length <= 1) {
            $results[$name] = $value;

        } else {
            $results[$name] = array();

            foreach ($item->childNodes as $subitem) {
                if($subitem->nodeType == 1) {
                    $name1 = $subitem->nodeName;
                    $value1 = $subitem->textContent;

                    $results[$name][$name1] = $value1;
                }
            }
        }
    }
}

// RESULTS MAY NOW BE HANDLED AS AN ARRAY
// $id_number = $results['id-number'];

?>
```

Now you should have the ID number of this request ($id), and an array containing the contents of the XML **<response>** key.   A finished implementation takes these values and uses them to determine whether or not to proceed with the transaction, and how to process the contents of the **<response>** key, such as **<result>**, **<summary-results>**, **<questions>**, **<velocity-result>,** and others.

Sample XML Responses

The contents of the result output are all enclosed in an XML tag called **<response>.** Furthermore, the first child tag is **<id-number>**. This provides the ID number which is used for reference, accounting or troubleshooting purposes.

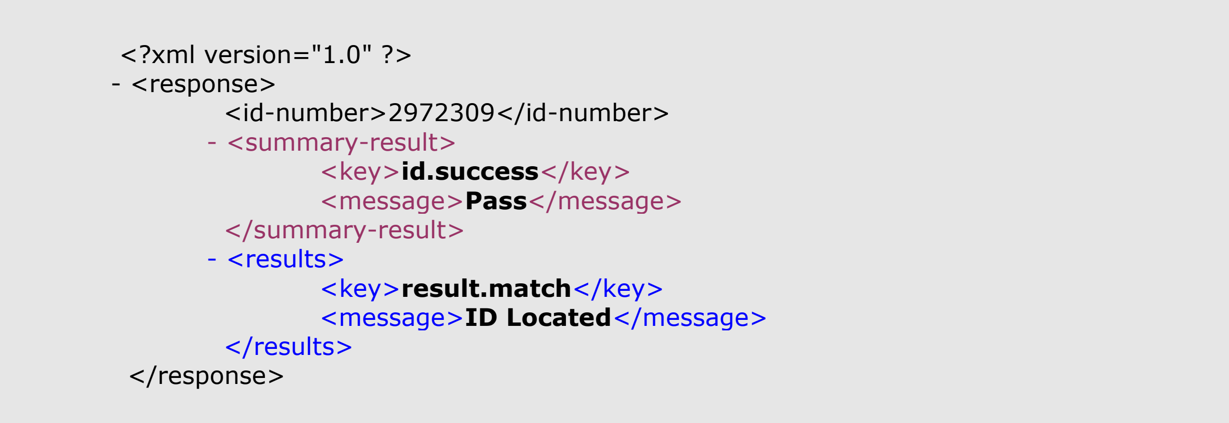
The **<results>** and **<summary-results>** are distinct, and should be used according to the Enterprise Configuration. For example, if Summary Results are enabled, **<summary-results>** should be used, and **<results>** should be ignored.

The **<summary-result>** (highlighted, purple) tag shows the ExpectID Summary Result, which will be one of three possible results: **id.success**, **id.failure**, or **id.partial**.

The **<results>** tag (highlighted, blue) shows the ExpectID result. In the non-configurable **<key>**, the result will either be **result.match**, **result.no.match,** or **result.match.restricted**.

The **<results>** value indicates if the information submitted matched the information located, subject to IDology's default tolerances and error-correcting logic (if enabled) on common typos in addresses or names.

Example: Subject is located

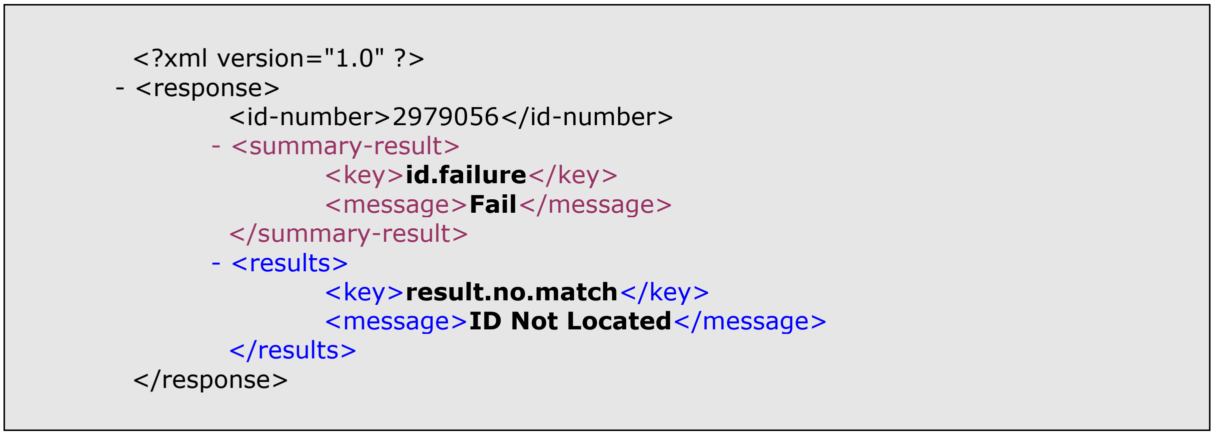


```
 <?xml version="1.0" ?>
- <response>
            <id-number>2972309</id-number>
          - <summary-result>
                      <key>id.success</key>
                      <message>Pass</message>
            </summary-result>
          - <results>
                      <key>result.match</key>
                      <message>ID Located</message>
            </results>
  </response>
```

Throughout the XML response, **<key>** and **<message>** pairs are returned through the API. The **<key>** value is the non-configurable output from the search and is what developers generally want to search. The **<message>** contents are the user-definable text corresponding to each of the three possible summary results. The user definable text is specified in the IDCenter and is used for configuring display messages in the ID Credential search tool or in your application.

ID Notes include warnings or informative messages (such as **resultcode.coppa.alert** for subjects who are age 13 or under, or **resultcode.subject.deceased** in the event that a subject's records indicate that they are dead.

Example: Subject is not located



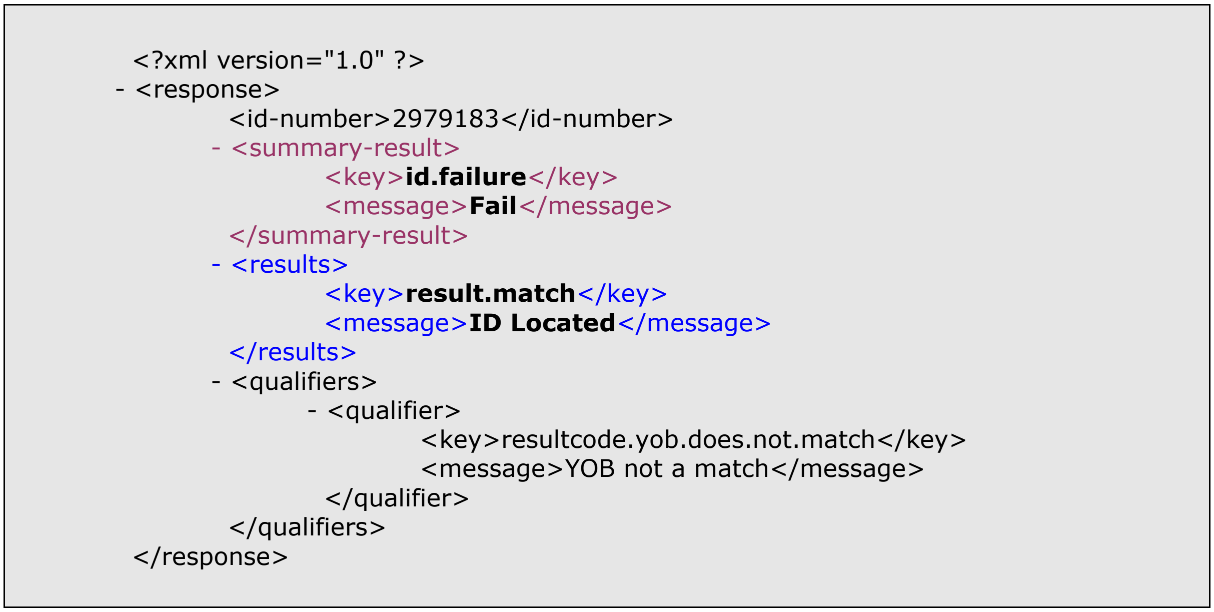
```
  <?xml version="1.0" ?>
- <response>
      <id-number>2979056</id-number>
    - <summary-result>
          <key>id.failure</key>
          <message>Fail</message>
      </summary-result>
    - <results>
          <key>result.no.match</key>
          <message>ID Not Located</message>
      </results>
  </response>
```

As mentioned above, the value returned via the **<summary-result>** tag is distinct from the value returned via the **<results>** tag. The **<summary-result>** value should only be used if your Enterprise configuration, viewable in the IDCenter has Summary Results enabled.

For each Enterprise, the Summary Results configuration in the IDCenter is a business decision, reflecting business policies and goals, and should not be considered part of the IDology integration. Using Summary Results allows an Enterprise to change its business rules in the IDCenter without having to complete any code changes.

The preceding examples show a basic consistency between **<results>** and **<summary-result>** but the results can differ, as the following example demonstrates. The **<summary-result>** values are based on customizable rules. The **<results>** are based on IDology's default logic. In the case below, the Summary Results configuration in the portal had the YOB Does Not Match rule entry set to "Fail". In the input for this example, the YOB was incorrect.

Example: <summary-result> and **<results>** are not the same



```
  <?xml version="1.0" ?>
- <response>
      <id-number>2979183</id-number>
    - <summary-result>
          <key>id.failure</key>
          <message>Fail</message>
      </summary-result>
    - <results>
          <key>result.match</key>
          <message>ID Located</message>
      </results>
    - <qualifiers>
        - <qualifier>
              <key>resultcode.yob.does.not.match</key>
              <message>YOB not a match</message>
          </qualifier>
      </qualifiers>
  </response>
```

Optional Differentiating Question

Sometimes the ExpectID process locates records on more than one individual and similarities in the information located making it impossible to identify the correct individual.

ExpectID Differentiator when enabled in the IDology ID Center is used to correct this situation. When ExpectID Differentiator is triggered, the API will return one question along with the ID results. The answer to the differentiating question identifies the correct record or records to use in the ExpectID or ExpectID IQ process.

Example: An ID result that includes an age-based differentiator question

```
-<response>
        <id-number>2543518</id-number>
        <summary-result>
                <key>id.partial</key>
                <message>Partial</message>
        </summary-result>
        <results>
                <key>result.match</key>
                 <message>ID Not Located</message>
        </results>
        <qualifiers>
                <qualifier>
                        <key>resultcode.multiple.records.found</key>
                        <message>Multiple records found</message>
                </qualifier>
        </qualifiers>
        <differentiator-question>
                <prompt>How old are you?</prompt>
                <type>age.range</type>
                <answer>29 - 38</answer>
                <answer>39 - 48</answer>
                <answer>49 - 58</answer>
                <answer>59 - 68</answer>
                <answer>None of the above</answer>
        </differentiator-question>
 </response>
```

As can be seen, the response includes an element called <differentiator-question>. The differentiating question does not depend on the value in the results element. The differentiator can be presented for both <key>**result.match**</key> and <key>**result.no.match**</key> outcomes. The differentiating question should not be discarded because of a <key>**result.no.match**</key> outcome from the initial search.

To answer the question, you must send an HTTP POST to either of the following urls:

https://web.idologylive.com/api/differentiator-answer.svc

https://web.idologylive.com/api/differentiator-answer-iq.svc

The differentiator-answer.svc is for ExpectID transactions only, and will not return ExpectID IQ questions. The differentiator-answer-iq.svc url must be used to submit answers for the differentiator question.

The request to post the response from the differentiating question must have all the following fields populated:

```
// ID FIELDS - PLEASE INCLUDE ALL FIELDS
$fields = array(
        'username'              => '',   // YOUR ExpectID USERNAME (16)
        'password'              => '',   // YOUR ExpectID PASSWORD
        'idNumber'              => '',   // IDOLOGY ID NUMBER FOR THE TRANSACTION
        'question1Type'         => '',   // <type> FOR THE DIFFERENTIATING QUESTION
        'question1Answer'       => '',   // <answer> FOR THE DIFFERENTIATING QUESTION
```

After the response to the differentiating question is submitted, the API will return standard ExpectID or ExpectID IQ results.

Example: XML results of correctly answered differentiating question returning ExpectID response

```
  <?xml version="1.0" ?>
- <response>
          <id-number>4610482</id-number>
          <summary-result>
                    <key>id.success</key>
                    <message>Pass</message>
          </summary-result>
          <results>
                    <key>result.match</key>
                    <message>ID Located</message>
          </results>
</response>
```

If the response to the Differentiating Question fails to identify the correct individual, the response looks like this:

Example: XML results of incorrectly answered differentiating question

```
  <?xml version="1.0" ?>
- <response>
            <id-number>4610501</id-number>
          - <summary-result>
                      <key>id.failure</key>
                      <message>Fail</message>
            </summary-result>
          - <results>
                      <key>result.no.match</key>
                      <message>ID Not Located</message>
            </results>
  </response>
```

Critical Errors

Important:

> For integrated applications, the two leading causes of loss of service are password changes for the API User Account, and IP Restrictions. IDology strongly recommends integrated applications be designed to alert the customer’s internal escalation process via email, pager, or the mechanism in use in the customer’s organization.

Password changes made for user ID’s, including API Accounts, using the IDCenter take effect immediately, and are not reversible. Unless the password is also updated in the integrated application, the API requests will begin failing immediately, and the XML response will look like this:

```
<response>
        <error>Invalid username and password</error>
</response>
```

If separate application instances are maintained, all instances of the application must have access to the API account password. IDology customers are required to protect the password in a secure database that is accessible only to the application and selected administrators.

External IP addresses can change without warning if any of the following happen:

- A network device is replaced
- Code is deployed to a new server
- The application infrastructure fails over to a different network device, or a different IP on the same device.
- A failover event occurs and the application is brought up in a redundant architecture

Whatever the cause for the IP address change, if the integrated application is making requests to IDology from an IP address that is not included in your company specific IP Whitelist, the request will be denied, and the XML response will be returned showing the reason.

```
<response>
        <error>Your IP address is not registered. Please call IDology Customer
        Service (866-520-1234).</error>
</response>
```

STEP 4 – Processing the XML Response when Questions are returned

If ExpectID IQ questions are part of an Enterprise configuration, they can be generated for each transaction, or they can be triggered by specific rules based on the results of a search, (e.g. the dollar amount of an associated transaction, from a triggered Velocity warning, or the detection of a restricted SKU or ZIP code)

When any of those configuration options exist and generate questions, they are presented along with the ExpectID results in the XML returned from the API in the **<questions>** tag.

Example: XML response, no questions or **<questions>** tag

```
<?xml version="1.0" ?>
- <response>
      <id-number>2979594</id-number>
    - <summary-result>
          <key>id.success</key>
          <message>Pass</message>
      </summary-result>
    - <results>
          <key>result.match</key>
          <message>ID Located</message>
      </results>
    - <qualifiers>
        - <qualifier>
              <key>resultcode.address.does.not.match</key>
              <message>Address Does Not Match</message>
          </qualifier>
      </qualifiers>
  </response>
```

The same input passed through the API can generate questions if any of the following are true:

- ExpectID IQ questions are enabled for all transactions
- Summary Results rules trigger questions
- ExpectID IQ input rules generate questions based on credential type and/or dollar amount associated with the order
- Velocity Monitors are triggered and are configured to generate questions
- Restricted SKUs are detected and the SKU monitor is configured to generate questions
- Restricted ZIP codes are detected and the ZIP restriction is configured to generate questions

Example: ExpectID IQ questions returned from a search

```
  <?xml version="1.0" ?>
- <response>
           <id-number>2979585</id-number>
         - <summary-result>
                     <key>id.success</key>
                     <message>Pass</message>
           </summary-result>
         - <results>
                     <key>result.match</key>
                     <message>ID Located</message>
           </results>
         - <qualifiers>
                   - <qualifier>
                               <key>resultcode.address.does.not.match</key>
                               <message>Address Does Not Match</message>
                     </qualifier>
           </qualifiers>
         - <questions>
                   - <question>
                               <prompt>Which of the following address numbers do you relate to the
                             residence associated with STILLWOOD DR?</prompt>
                               <type>street.number</type>
                               <answer>1333</answer>
                               <answer>1212</answer>
                               <answer>8629</answer>
                               <answer>518</answer>
                               <answer>1811</answer>
                               <answer>None of the above</answer>
                       </question>
                   - <question>
                               <prompt>When you were associated with the city of ATLANTA, in which
                             of the counties below was that location?</prompt>
                               <type>county</type>
                               <answer>HUNTERDON</answer>
                               <answer>DEKALB</answer>
                               <answer>CONECUH</answer>
                               <answer>DE SOTO</answer>
                               <answer>MARSHALL</answer>
                               <answer>None of the above</answer>
                       </question>
                   - <question>
                               <prompt>When you were associated with the residence on CASTLE
                             FALLS DR, in which of the cities below was this address?</prompt>
                               <type>city.of.residence</type>
                               <answer>CANYONDAM</answer>
                               <answer>ASPEN</answer>
                               <answer>PIMA</answer>
                               <answer>LUPTON</answer>
                               <answer>ATLANTA</answer>
                               <answer>None of the above</answer>
                     </question>
           </questions>
  </response>
```

Optional Skip Question:

When enabled, this option allows one ExpectID IQ question to be skipped and replaced with another question. This works by supplying one additional question in the XML response that can be used in place of the skipped question.

It is important to note that this feature is feasible only for customers who present questions one at a time. This feature is not supported where questions are presented all at once.

In the XML response below, the ExpectID IQ settings in the IDCenter are configured to return 2 questions with the “Skip Question” option enabled. When the “Skip Question” option is enabled, one extra question (in bold type in the example below) is delivered in the XML response in the event a consumer opts to skip a question and an additional possible response is generated for each question which is “Skip this question.”

If a question is skipped, the response to the question should be sent along with the other responses. For example, if two questions are specified in the IDology IDCenter with the option to skip a question selected, three questions will be delivered. In this example, the IDology API would still expect 2 responses. Therefore, you are only required to return answers for the number of questions configured for your enterprise. However, in order to successfully complete the question set, you will need to submit two “real” answers.

Example: Your enterprise is configured to present a question set with 2 questions and the Skip Question feature is enabled. IDology API sends three questions and you return the answers for any two of them. The answer for the third question, or “skipped” question will be set to "not presented". If you return one skipped response, and one real response, the result will be "incomplete" because the IDology API did not receive two real answers for the question set.

In addition, you will need to ensure that you eliminate the skip option once the customer has skipped a question. For example in the two question set referenced above, if the customer skips question one, then “Skip Question” cannot be presented as an option for question two.

Example: ExpectID IQ Skip Question Enabled

```
<?xml version="1.0" ?>
- <response>
  <id-number>55229</id-number>
- <summary-result>
  <key>id.success</key>
  <message>Match Found</message>
  </summary-result>
- <results>
  <key>result.match</key>
  <message>ID Located</message>
  </results>
- <questions>
        - <question>
                <prompt>In which city is STILLWOOD DR?</prompt>
                <type>city.of.residence</type>
                <answer>PARSHALL</answer>
                <answer>KENOSHA</answer>
                <answer>LAKE VILLAGE</answer>
                <answer>KARVAL</answer>
                <answer>LUPTON</answer>
                <answer>None of the above</answer>
                <answer>Skip the question</answer>
```

```
     </question>
    - <question>
          <prompt>In which month were you born?</prompt>
          <type>month.of.birth</type>
          <answer>May</answer>
          <answer>July</answer>
          <answer>September</answer>
          <answer>October</answer>
          <answer>January</answer>
          <answer>None of the above</answer>
          <answer>Skip the question</answer>
     </question>
    - <question>
          <prompt>Where was your social security number issued?</prompt>
          <type>ssn.issued.in</type>
          <answer>North Carolina</answer>
          <answer>Utah</answer>
          <answer>New Jersey</answer>
          <answer>Georgia</answer>
          <answer>Tennessee</answer>
          <answer>None of the above</answer>
          <answer>Skip the question</answer>
     </question>
</questions>
</response>
```

STEP 5 – Submitting the Answers

Preparing to Submit Answers

Once questions are extracted from the ExpectID XML response you are able to present them to the subject.

Keep track of the **<answer>** that corresponds to the **<type>** for each question and send them back to IDology so they can be compared to the correct answers and your ExpectID IQ result can be generated. Even if you present the questions to the subject one at a time, you must submit them to IDology all at once.

Submit Answers

Submit answers for ExpectID IQ to the following URL:

https://web.idologylive.com/api/idliveq-answers.svc .

The following parameters must be included (all fields are required)

```
// ID FIELDS - PLEASE INCLUDE ALL FIELDS (EVEN IF BLANK)
$fields = array(
      'username'               => '',  // YOUR ExpectID USERNAME (16)
      'password'               => '',  // YOUR ExpectID PASSWORD
      'idNumber'               => '',  // IDOLOGY ID NUMBER FOR THE TRANSACTION
      'question1Type'          => '',  // <type> FOR THE FIRST QUESTION
      'question1Answer'        => '',  // <answer> FOR THE FIRST QUESTION
      'question2Type'          => '',  // <type> FOR THE SECOND QUESTION
      'question2Answer'        => '',  // <answer> FOR THE SECOND QUESTION
      'question3Type'          => '',  // <type> FOR THE SECOND QUESTION
      'question3Answer'        => '',  // <answer> FOR THE SECOND QUESTION
```

The order of the questions is not relevant. What matters is that you properly pair the question types with the question answers and that you include the answers for all three of the questions that you retrieved and presented to the subject.

The examples here show 3 questions, but there may be anywhere from 1 to 5 questions in the question set, given the Enterprise configuration rules set in the IDCenter.

*Note: There are some questions which will always return a specific number of answers, regardless of the above setting. Questions ID38, ID39, ID46, ID47, ID48 and ID49 will always return 4 answers. Questions ID31, ID32, and ID33 will always return 6 answers.

The Answers Response

When the subject's answers are submitted properly, the result response is like the following:

```
  <?xml version="1.0" ?>
- <response>
          <id-number>2979585</id-number>
        - <summary-result>
                    <key>id.success</key>
                    <message>Pass</message>
          </summary-result>
        - <results>
                    <key>result.match</key>
                    <message>ID Verified</message>
          </results>
        - <qualifiers>
                  - <qualifier>
                    <key>resultcode.address.does.not.match</key>
                    <message>Address Does Not Match</message>
                    </qualifier>
          </qualifiers>
          <answers-received>3</answers-received>
        - <idliveq-result>
                    <key>result.questions.1.incorrect</key>
                    <message>One Incorrect Answer</message>
          </idliveq-result>
          <iq-summary-result>pass</iq-summary-result>
  </response>
```

Once again, the original ExpectID results are included for convenience. The number of answers received is also included for debugging purposes, in **<answers-received>**. If a different number of answers are received than questions are issued, or if invalid question types are returned, the value in <idliveq-result> will be **result.incomplete.**

```
<?xml version="1.0" ?>
- <response>
        <id-number>2979585</id-number>
        - <summary-result>
                <key>id.success</key>
                <message>Pass</message>
        </summary-result>
        - <results>
                <key>result.match</key>
                <message>ID Located</message>
        </results>
        <answers-received>1</answers-received>
        - <idliveq-result>
                <key>result.incomplete</key>
                <message>result.incomplete</message>
        </idliveq-result>
</response>
```

Finally, the actual ExpectID result is presented in **<iq-summary-result>** as defined by the Pass/Fail configuration made by your Administrator

At this point, the ExpectID IQ transaction is complete. You can use the IDology reports to view the results of the ID, as well as the questions generated and the answers submitted.

If ExpectID Challenge Questions are enabled, a separate request must be made to the URL for Challenge Questions. Challenge Questions are handled in much of the same way as regular ExpectID IQ questions. *(Refer to the ExpectID Challenge API Guide which covers Challenge Questions in more detail.)*

Errors

There are two main errors associated with ExpectID™ IQ requests. The first occurs if you attempt to retrieve questions for an ID that is not eligible for questions. In that case, the following result will be returned:

```
<response>
   <id-number>116</id-number>
   <summary-result>
      <key>id.success</key>
      <message>Pass</message>
   </summary-result>
   <results>
      <key>result.match</key>
      <message>ID Located</message>
   </results>
   <idliveq-error>
      <key>id.not.eligible.for.questions</key>
      <message>Not eligible for questions.</message>
   </idliveq-error>
</response>
```

When you submit a questions request, you should look for the idliveq-error element if you can’t locate any questions. The other error that’s possible occurs if we cannot generate questions for a particular ID. The response for that type of ID is as follows:

```
<response>
   <id-number>116</id-number>
   <summary-result>
      <key>id.success</key>
      <message>Pass</message>
   </summary-result>
   <results>
      <key>result.match</key>
      <message>ID Located</message>
   </results>
   <idliveq-error>
      <key>result.questions.no.data</key>
      <message>Unable to generate questions</message>
   </idliveq-error>
</response>
```

If for some reason the service is down, this message is displayed:

```
 <response>
    <failed>Service temporarily unreachable</failed>
</response>
```

Generally speaking, if you receive the <error> response it indicates that there’s a problem with way your integration code is submitting requests. If you received the <failed> response it means that there’s a problem with the service. Contact IDology support for any questions related to error messages.

Example: ID Number in answer-submission is malformed (in this case, the last digit was left off):

```
<?xml version="1.0" ?>
- <response>
          <failed>Exception initializing proxy: [com.idology.persist.Query#297958]</failed>
  </response>
```

Example: Answers already returned:

```
<?xml version="1.0" ?>
- <response>
          <error>Questions already answered for this ID</error>
  </response>
```

Example: Bad password, or invalid username.

```
<?xml version="1.0" ?>
- <response>
          <error>Invalid username and password</error>
  </response>
```

Example: Invalid Query ID

```
<?xml version="1.0" ?>
- <response>
          <error>Invalid query ID</error>
  </response>
```

PHP Sample Code for Answer Submission and Response

```
<?php

// YOUR IDOLOGY ACCOUNT INFORMATION
$account  = '&username=USERNAME';
$account .= '&password=PASSWORD';

// ID FIELDS - PLEASE INCLUDE ALL FIELDS
$fields = array(
'idNumber'        => '',
'question1Type'   => '',
'question1Answer' => '',
'question2Type'   => '',
'question2Answer' => '',
'question3Type'   => '',
'question3Answer' => ''
);

// CONNECT TO IDOLOGYLIVE,
// SUBMIT ID REQUEST AND READ RESULTS

$url = "https://web.idologylive.com";

$query = $account;
foreach ($fields as $field => $value) {
        $query .= '&' . $field . '=' . urlencode($value);
}

$fp = pfsockopen($url, 443, $errno, $errstr);

$content = "";

if (!$fp) {
   echo "$errstr ($errno)<br/>\n";
} else {
   fputs($fp, "POST /api/idliveq-answers.svc  HTTP/1.0\r\n");
   fputs($fp, "Host: web.idologylive.com\r\n");
   fputs($fp, "Content-type: application/x-www-form-urlencoded\r\n");
   fputs($fp, "User-Agent: Mozilla 4.0\r\n");
   fputs($fp, "Content-length: ".strlen($query)."\r\n");
   fputs($fp, "Connection: close\r\n\r\n");
   fputs($fp, $query);

   do {
       $data = @fread($fp, 8192);
       if (strlen($data) == 0) {
           break;
       }
       $content .= $data;
   } while(true);

   @fclose($fp);
}

// STRIP HTTP HEADERS
$header_pos = strpos($content, "\r\n\r\n") + 4;
$content = substr($content, $header_pos);
```

Other XML Output

In the course of processing responses, other tags may appear:

<failed> Provides descriptive text should the ID request fail.

<error> Indicates any problems with the account or submitted data with a description of the error.

<restriction> Includes details in the event that a subject matches a user-defined restriction. Can include either SKU Restrictions, ZIP Code Restrictions, and ExpectID PA Alerts.

<velocity_results> Includes details in the event that a transaction has triggered user-defined velocity monitor

Velocity

IDology's Velocity Monitoring system is a powerful anti-fraud feature. It monitors incoming or located credentials for recurrence, either for a single enterprise or across the entire IDology portfolio of Enterprises. It is common for IDology clients to have more than one velocity monitor configured; some monitors look for Address only, others for combinations of Name, Address, SSN4, Phone Number, and other attributes.

Any monitor can be combined with a total dollar amount for orders during a selected timeframe, or can be configured to trigger when a dollar amount exceeds a specified amount.

Monitors can also be set for a particular time period up to 14 days, and can also be set to monitor a specified hour time range within each day.

Configuring velocity monitors is almost always the responsibility of Loss-Prevention or Finance officers.

In this example, two velocity events occurred.

```
<?xml version="1.0" ?>
- <response>
      <id-number>2980120</id-number>
    - <summary-result>
          <key>id.success</key>
          <message>Pass</message>
      </summary-result>
    - <results>
          <key>result.match</key>
          <message>ID Located</message>
      </results>
    - <velocity-results>
        - <velocity-result>
              <key>ADDR</key>
              <message>Velocity Warning - Address Only, 12 hours</message>
          </velocity-result>
        - <velocity-result>
              <key>NAME</key>
              <message>Velocity Warning - Name and Address, 12 hours</message>
```

```
            </velocity-result>
        </velocity-results>
</response>
```

Possible Results for Velocity

Result	What it means
- <velocity-results> - <velocity-result> <key>NAME2</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Located First Name and Last Name per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>LNAME4</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Located Last Name and SSN4 per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>NAME</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Located Last Name and Street Address per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>NAME4</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Located Last Name and Street Address w/ SSN4 per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>ADDR</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Located Street Address per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>STRNZIP</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Located Street Name and ZIP Code per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>ADDR4</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Located Street Address w/SSN4 per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>PHONE</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Located Phone Number per your enterprise configuration.

- <velocity-results> - <velocity-result> <key>SSN9</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Located SSN9 per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>INAME2</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Input First Name and Last Name per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>ILNAME4</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Input Last Name and SSN4 per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>INAME</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Input Last Name and Street Address per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>INAME4</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Input Last Name and Street Address w/ SSN4 per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>IADDR</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Input Street Address per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>IALTADDR</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on the Alternate Street Address per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>ISTRNZIP</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Input Street Name and ZIP Code per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>IADDR4</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Input street address w/ SSN4 per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>IEMAIL</key>	A Velocity alert was triggered on Input Email Address per your enterprise configuration.

<message>Your custom text appears here</message> </velocity-result> </velocity-results>	
- <velocity-results> - <velocity-result> <key>IPHONE</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Input Phone Number per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>ISSN9</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on Input SSN9 per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>IIPADDR</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered in Input IP Address per your enterprise configuration.
- <velocity-results> - <velocity-result> <key>MAC</key> <message>Your custom text appears here</message> </velocity-result> </velocity-results>	A Velocity alert was triggered on the Input MAC Address per your enterprise configuration.

ExpectID Score

IDology's ExpectID Score is a great analytical feature that allows clients to gauge their risk level of a transaction. This feature is fully customizable and configurable by the client. The client may choose to use ExpectID Score to just monitor the transaction and/or make decisions based on low, medium or high risk transactions.

When enabled, the <idnotescore> tag should be searched for in XML results.

```
<response>
      <id-number>1108681053</id-number>
      <summary-result>
            <key>id.success</key>
            <message>Pass</message>
      </summary-result>
      <results>
            <key>result.match</key>
            <message>ID Located</message>
      </results>
      <qualifiers>
            <qualifier>
                  <key>resultcode.address.does.not.match</key>
                  <message>Address Does Not Match</message>
            </qualifier>
            <qualifier>
                  <key>resultcode.street.number.does.not.match</key>
                  <message>Street Number Does Not Match</message>
            </qualifier>
            <qualifier>
                  <key>resultcode.street.name.does.not.match</key>
                  <message>Street Name Does Not Match</message>
            </qualifier>
            <qualifier>
                  <key> resultcode.low.risk </key>
                  <message>Low Risk Score</message>
            </qualifier>

      </qualifiers>
      <idnotescore>123</idnotescore>
</response>
```

Full List of ID Notes

Each IDology client has different policies, rules, and guidelines regarding the different ID Notes. Some clients are not concerned about the contents of ID Notes, and focus solely on the <results> that are returned in XML results.

The following list includes all ID Notes, which are enclosed in the <qualifiers> key as <qualifier> subkeys, that may be returned.

resultcode.coppa.alert	Subject is 13 or under. COPPA laws forbid conducting e-commerce with people under 14 years of age. Due to the issues around storing data on children, when this result is encountered, the input information and located data will not be available in any record of the transaction.
resultcode.address.does.not.match	Address found does not match address submitted. This can be due to a typo in the input information, typos or errors in the address located, or when the address is actually incorrect, but the subject's credentials are located in or near the target ZIP code, city, or metropolitan area.
resultcode.street.name.does.not.match	The submitted street name does not match the located data. This can be due to a typo in the input or the located data. This note will always be accompanied by the resultcode.address.does.not.match ID Note.
resultcode.street.number.does.not.match	The submitted street number does not match the located data. This can be due to a typo in the input or the located data. This note will always be accompanied by the resultcode.address.does.not.match ID Note.
resultcode.input.address.is.po.box	The input address is a PO Box. This does not indicate any discrepancy between the input address and the located data.
resultcode.warm.address.alert	This indicates the located address is a warm address. Warm addresses will be clarified further as to which type with the value of the <warm-address-list> value. Those are: • mail drop • hospital • hotel • prison • campground • college / university • USPO
resultcode.zip.does.not.match	ZIP code located does not match the ZIP code submitted.
resultcode.dob.does.not.match	DOB found does not match DOB submitted. This can be due to errors in the located data, and does not necessarily mean the ID Located is not valid, especially when the SSN4 was also provided. Numerous public record data sources do not include birthdate information in their records.
resultcode.yob.does.not.match	The year of birth located does not match. If neither the month-of-birth nor the year-of-birth

	match, this ID Note is presented.
resultcode.yob.within.one.year	This indicates a discrepancy between the YOB submitted and the YOB located of 1 year. If the YOB submitted is 1970 and the YOB located is 1971, the ID Note presented.
resultcode.mob.does.not.match	This indicates a discrepancy between the month of birth submitted and the month of birth located. This ID Note is only presented when the year-of-birth matches.
resultcode.no.mob.available	This indicates no month-of-birth was included in the records that located.
resultcode.multiple.records.found	Several valid records exist containing conflicting identifying information.
resultcode.newer.record.found	The subject of the search was found at the address submitted, but a more recent record shows a different address for the individual. If a subject intentionally provides an old address and a new one is located, this will happen, but it also results from a subject providing a current "home" address and the IDology search locating newer real estate, like a second/vacation home or an investment property.
resultcode.high.risk.address.alert	Identifies addresses with a known history of fraud activity.
resultcode.address.velocity.alert	Warns of the number of addresses someone has had within a defined time period.
resultcode.address.stability.alert	Indicates how often someone moves based on specific timeframe triggers
resultcode.address.longevity.alert	Specifies how long someone has lived at their current address.
resultcode.address.location.alert	A location-based alert that advises you when the located ZIP Code exceeds your Enterprise's Permitted Distance Radius rule.
resultcode.alternate.address.alert	Indicates the Alternate Address could not be verified for the consumer. The Alert will be clarified further as to which type with the value of the <alternate-address-list> value. Those are: • Street Number • Street Name • State • ZIP Code
resultcode.no.dob.available	The individual was identified, but DOB information was not available in the records located. This does not mean the search failed. Numerous public-record data sources do not include DOB information in their records.
resultcode.ssn.not.available	The individual was identified, but SSN information was not available. This does not mean the search failed. Numerous public-record data sources do not include SSN information in their records.
resultcode.ssn.does.not.match	SSN found does not match SSN submitted. This does not necessarily mean the ID Located is invalid, especially when the MOB+YOB or YOB was provided as well. There can be errors in the

	SSN data that is located.
resultcode.ssn.within.one.digit	This indicates a discrepancy between the SSN submitted and the SSN located of 1 digit. If the SSN submitted is off by one digit of the located SSN, the ID Note is presented.
resultcode.input.ssn.is.itin	Indicates the Input SSN is an ITIN (Individual Taxpayer Identification Number).
resultcode.located.itin	Indicates the Located SSN is an ITIN (Individual Taxpayer Identification Number).
resultcode.subject.deceased	Records indicate that the subject in question is deceased.
resultcode.state.does.not.match	This indicates the State located does not match the state input as part of the address. This IDNote is a very specialized response, and is only available if configured by IDology.
resultcode.ssn.issued.prior.to.dob	This indicates the SSN number was issued before the individual's DOB, a serious fraud flag.
resultcode.ssn.not.valid	The located SSN does not match the structure of a valid SSN.
resultcode.single.address.in.file	This indicates only a single address record was located for the individual.
resultcode.data.strength.alert	Examines how long someone has been in the public record system and how much data is available.
resultcode.last.name.does.not.match	This indicates that the located last name does not match the input last name.
resultcode.thin.file	This indicates the record located had very little information, specifically only name + address (Personal Info or "PI" only), and lacks any information that can be used to link to other records.
resultcode.bankruptcy	This indicates that the subject of the search has a chapter 7 or 13 bankruptcy in their public record.
resultcode.age.below.minimum	Indicates that the subject is below the minimum age specified in the enterprise configuration.
resultcode.age.above.maximum	Indicates that the subject is above the maximum age specified in the enterprise configuration.
resultcode.multiple.records.found	Indicates more than one record was found containing the same located address for the person but one of the records contains a different SSN than the others.
resultcode.blacklist.alert.ssn	Indicates the input SSN9 was found on the Alert List for the Enterprise.
resultcode.blacklist.alert.address	Indicates the input Address was found on the Alert List for the Enterprise.
resultcode.blacklist.alert.ip	Indicates the input IP Address was found on the Alert List for the Enterprise.
resultcode.blacklist.alert.phone	Indicates the input Phone Number was found on the Alert List for the Enterprise.
resultcode.blacklist.alert.email	Indicates the input Email Address was found on the Alert List for the Enterprise.
resultcode.blacklist.alert.domain	Indicates the input Email Domain was found on the Alert List for the enterprise.

resultcode.blacklist.alert.mac.address	Indicates that input MAC Address was found on the Alert List for the enterprise
resultcode.restricted.ip.proxy <ip-address-type>anonymous</ip-address-type>	Indicates an active anonymous proxy which the person may be using to deliberately hide their true location.
resultcode.restricted.ip.proxy <ip-address-type>satellite</ip-address-type>	Indicates a person is accessing the Internet through a satellite provider.
resultcode.restricted.ip.proxy <ip-address-type>mobile gateway</ip-address-type>	Indicates a person is using a mobile gateway proxy and connecting to the Internet through a mobile device.
resultcode.restricted.ip.proxy <ip-address-type>regional</ip-address-type>	Indicates a regional proxy is being used to route traffic from multiple states within a single country.
resultcode.restricted.ip.country	The IP address is found to be from a country that is set as restricted within the Enterprise configuration.
resultcode.ip.location.alert	Indicates the IP address is not in the same state nor is it within proximity to the located address for the person.
resultcode.ip.state.does.not.match	Indicates the located IP State does not match the located State for the consumer.
resultcode.ip.invalid	Indicates the IP address submitted does not fit the proper structure of an IP address and/or is found to be an unassigned IP address. This might also indicate that the IP address is a private or multicast address.
resultcode.ip.not.located	Indicates the IP address could not be located within our data sources.
resultcode.ip.location.not.**a**vailable	Indicates the location of the IP address cannot be determined.
resultcode.high.risk.phone.suspicious	Indicates that the Phone Number is considered suspicious within our high risk data sources.
resultcode.high.risk.phone.fraud	Indicates that the Phone Number is associated with fraudulent activity within our high risk data sources.
resultcode.low.risk	Triggered when the total ID Score is less than or equal the value entered for the Low Risk threshold.
resultcode.medium.risk	Triggered when the total ID Score is between the values entered for the Low and High Risk thresholds.
resultcode.high.risk	Triggered when the total ID Score is greater than or equal to the value entered for the High Risk threshold. .

Step 6 – Input Considerations

One key to maximizing the results of IDology's service is doing as much as possible to get clean input from the end user. Input from end users, often in the name or address fields, is a frequent source of "problem input" that can lead to unsuccessful searches.

IDology's interface is designed to accept information in the form most suited to public record data searches. With respect to an individual's name, the search routines in IDology's system look strictly for first and last names, using legal given names as they appear on government issued ID. IDology's search logic does not use any of the following.

- Company or Business names
- "Husband and Wife" or "pairs" first names like "Don and Mary"
- Middle names or initials,
- Name prefixes like Mrs. or Dr.
- Name suffixes like Jr., Sr., PhD, MD, etc.

Addresses are treated similarly. IDology's search processes use street number and street name, and do not make use of

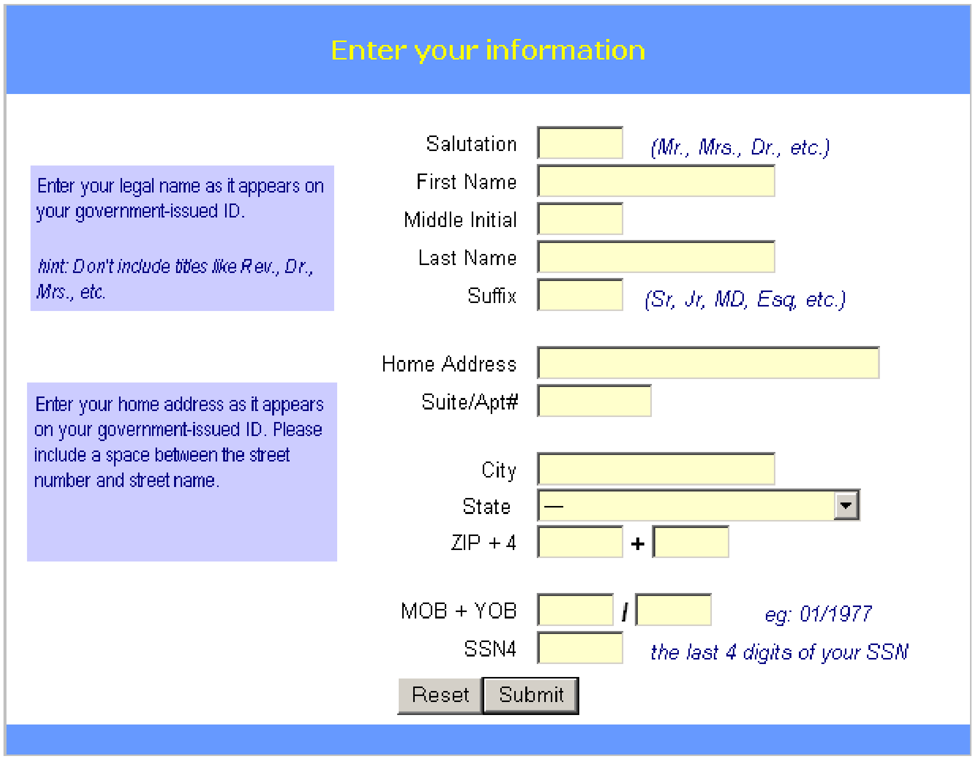
- Business addresses
- Directionals; 1000 Harbour St. West should be entered as 1000 Harbour St.
- Apartment, Suite or Unit numbers.

A successful strategy to handle this is to design applications to isolate some of this problem data, where possible. A good example is using an explicit middle name or middle initial field in a data-entry interface. Even though IDology doesn't use middle name or middle initial, the use of the field in the application reduces the likelihood that a middle initial or middle name is added to the first name field when the HTTPS POST to IDology's API is made.

Apartment numbers can be treated the same way. When the call to IDology is made, the apartment number input is simply omitted from the call to IDology's API.

Example - Isolating unwanted data like middle initial and apartment number in fields that should not be sent through the API.

Sample web page:



Enter your information

Enter your legal name as it appears on your government-issued ID.

hint: Don't include titles like Rev., Dr., Mrs., etc.

Enter your home address as it appears on your government-issued ID. Please include a space between the street number and street name.

Several fields in this form do not have a corresponding field in IDology's API, for instance middle initial and +4 ZIP. The objective in explicitly separating these fields is that unwanted data is isolated and can then be omitted when the call to IDology is made.

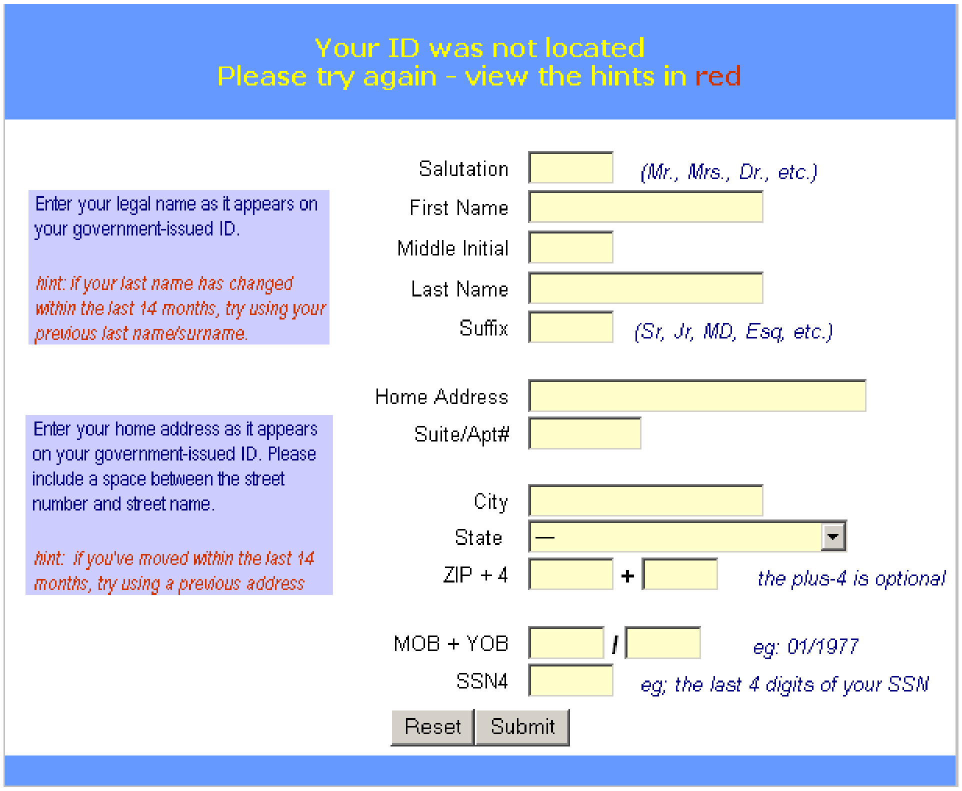
Even when a call is made to IDology with clean data, there may be unwanted results. The results (i.e. – the contents of the **<results><key>** tag) can return **result.no.match** or undesirable items in the **<qualifiers>** element.

In such cases, a good initial response is to display updated messaging in the data-entry application asking a remote end user to clarify their information and try again. Generally, what works best is to follow, or approximate the following requests for clarification, in the order listed here:

1. Ask the end user to verify that the name and address information were correctly entered. They key to entering a name correctly is that the name has to be a consumer's <u>legal given name as it appears on their government issued ID</u>. The address should also be entered as it appears on their government issued ID, without directionals or apartment numbers.
2. Prompt the consumer for a prior address. Example - "Have you moved or changed addresses within the last 12 months? If so, please enter your previous address"
3. Prompt the consumer for a prior surname. Example – "Have you changed your name within the last 18 months? If so, please try entering your previous name."

Example – Messaging to help an end user clarify their input on a second or third chance after getting an unsatisfactory response from the initial ID search.

Sample web page with message responses:



Your ID was not located
Please try again - view the hints in red

Enter your legal name as it appears on your government-issued ID.

hint: if your last name has changed within the last 14 months, try using your previous last name/surname.

Enter your home address as it appears on your government-issued ID. Please include a space between the street number and street name.

hint: if you've moved within the last 14 months, try using a previous address

Contacting IDology Customer Service

To contact IDology's customer service team you can:

1. Fill out the help request form located in the Resource Center
2. Call 866-520-1234
3. Email customerservice@idology.com.