iDetic Solutions

INTERFACE GUIDE

Version 1.0.16

iDeticSolutions.com | 1-866-4IDETIC | info@ideticsolutions.com

CONFIDENTIALITY WARNING

Except as required by law or with the prior written consent of iDetic Solutions, this document and all information contained herein shall be kept confidential by the parties hereto and their Representatives, and shall not be disclosed to any other person and shall be used only for the purposes provided herein.

Contents

1.0 Overview

The iDetic Query feature allows a user to perform an investigation on 100 samples all at once, saving countless hours of tedious labor all while stopping exceptions and other issues in real time.

To accomplish this iDetic must be able to communicate with our client's internal systems. With security and versatility at the top of our priorities we use either a highly secured and industry standard REST Web API or a system of file drops that acts as a middleman between our two systems. This allows us to communicate without creating invasive security or data integrity risks.

This document is intended to share the specific output/input format requirements between our systems. Please note that though the client may build their own API host from scratch, we can also provide a prebuilt template for most scenarios. Additionally, if your organization does not have the bandwidth or desire to setup the API or file drop interfaces, iDetic offers a service to configure and implement on your behalf.

2.0 API Interface

2.1 From iDetic to Client

Once an iDetic rack has been scanned, the user will be able to select a preconfigured QueryType and then hit the “Query” button in the iDetic web user interface. This action will trigger the iDetic software to post the Sample ID’s across the REST Web API in the below format. Note that if a client has chosen to have multiple QueryTypes configured it is a likely indication that different query criteria should be used based on the QueryType received from iDetic. Please verify the requirements for each QueryType with your sponsor.


```
{
  "selectedQuery":"QueryType",
   "sampleIds": [
      "VAL00000001",
      "VAL00000002"
      "VAL00000003",
      "VAL00000004",
      "VAL00000005",
      "VAL00000006",
      "VAL00000007",
      "VAL00000008",
      "VAL00000009",
      "VAL00000010"
   ]
}
```


*Note: The input above was truncated for space and readability, the REST Web API should allow up to 100 sample ids per request.

2.2 From Client to iDetic

Once you have received the ID's you may query them against your systems and retrieve/return any information your organization finds valuable. By default, there are 4 fields to send information back through: "TestRequests", "TestCompleted", "TestPending", and "TestFlags". These fields can be renamed by request.

Please return the data back through the API in the following format:

```
{
  "VAL00000001": {
    "TestRequests": [
      "HBS",
      "HCV",
      "HIV",
      "ZIKA"
    ],
    "TestCompleted": [
      "HBS",
      "HCV",
      "HIV"
    ],
    "TestsPending": [
      "ZIKA"
    ],
    "TestFlags": [
                {
                "Tests": [
                        "HIV"
                                ],
                                "TestFlags": ", Centrifugation Time Greater Than 24 Hours"
                }
        ]   },
  "VAL00000002": {
    "TestRequests": [
      "HBS",
      "HCV",
      "HIV"
    ],
    "TestCompleted": [
      "HBS",
      "HCV",
      "HIV"
    ],
    "TestsPending": null,
    "TestFlags": null
  }
}
```

2.3 Postman Settings

You may use the following screenshot as an example as a guide when configuring postman during initial setup and validation.

```
POST https://www [redacted] /api/Data/GetTests
POST [redacted] api/Data/GetTests HTTP/1.1
Authorization: Bearer [redacted]
nO8Df51uic
Content-Type: application/json
User-Agent: PostmanRuntime/7.20.1
Accept: */*
Cache-Control: no-cache
Postman-Token: d941a329-c86c-41f7-b7ee-62c46c768292
Host: www [redacted] net
Accept-Encoding: gzip, deflate
Content-Length: 43
Connection: keep-alive

{ "sampleIds":[ "3130361501","3830493962"]}

HTTP/1.1 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-Type: application/json; charset=utf-8
Expires: -1
Server: Microsoft-IIS/8.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
X-Powered-By: ARR/3.0
X-Powered-By: ASP.NET
Date: Fri, 31 Jan 2020 04:38:15 GMT
Content-Length: 247

{"3130361501":{"TestRequests":["HBS","HCV","HIV"],"TestCompleted":["HBS","HCV","HIV"],"TestsPending":null,"TestFlags":null},"3830493962":
```

3.0 File Interface

3.1 From iDetic to Client

Once an iDetic rack has been scanned, the user will be able to select a preconfigured QueryType and then hit the “Query” button in the iDetic web user interface. This action will trigger the iDetic software to create a JSON Query file at a configured network input location with the below filename and file format. Note that if a client has chosen to have multiple QueryTypes configured it is a likely indication that different query criteria should be used based on the QueryType received from iDetic. Please verify the requirements for each QueryType with your sponsor.

File Name: StationName_MMDDYYYYhhmmssfff.json

File Name Part	Value
Station Name	iDetic Station Name
MM	Two Digit Numerical Month Value
DD	Two Digit Numerical Day of Month Value
YYYY	Four Digit Numerical Year Value
hh	Two Digit Numerical Hour Value
mm	Two Digit Numerical Minute Value
ss	Two Digit Numerical Second Value
fff	Three Digit Numerical Millisecond Value

File Content Format:

```
{
  "selectedQuery":"QueryType",
  "sampleIds":[
    "SampleId1",
    "SampleId2",
    "SampleId3",
    "…",
    "…",
    "SampleId100"]
}
```

File Sample:
https://ideticsolutions.com/Documentation/STATIONNAME_04232020113825120.json

Notes: *The File Content above was truncated for space and readability, the query should allow up to 100 sample ids per request as shown in the File Sample.*

Please also note that if iDetic encounters an error at this step, the process will continue to step 3 and instead the error information will be returned to iDetic. This Error Handling will be handled by iDetic and no query file will be executed. This will avoid triggering the client query if an error is encountered during the iDetic process.

3.2 From Client to iDetic

The client system will need to continuously monitor the configured network input location for new json query files. Once a new file is found, the client system will pick up the new file, read its contents, remember its file name, delete the file, and perform a query with the parameters specified in the file contents. Once the query results are obtained, the client system must create a JSON Query Results file which will **ONLY include found sampleIds** at a configured output location with the below filename and file format. Note that the file name **must match the input file name,** except with _Results or _DumpResults appended to the end of the file name. Use _DumpResults only when the QueryType is “Dump”. Dump queries go through a separate process and thus require a separate file naming convention. The file format is otherwise the same.

File Name: [InputFileName]_Results.json

File Name: [InputFileName]_DumpResults.json

Standard File Sample:
https://ideticsolutions.com/Documentation/STATIONNAME_04232020113825120_Results.json

Dump File Sample:
https://ideticsolutions.com/Documentation/STATIONNAME_04232020113825120_DumpResults.json

File Content Format:


```
{
        "SampleId1":{
          "TestRequests":[
            "TestA",
            "…",
            "TestZ"
        ],
          "TestCompleted":[
            " TestA ",
            "…",
            " TestZ "
        ],
          "TestsPending":[
            " TestA ",
            "…",
            " TestZ "
        ],
          "TestFlags":null
        },
          "SampleId100":{
          "TestRequests":[
            " TestA ",
            "…",
            " TestZ "
        ],
          "TestCompleted":[
            " TestA ",
            "…",
            " TestZ "
        ],
          "TestsPending":[
            " TestA ",
            "…",
            " TestZ "
        ],
          "TestFlags":null
        }
}
```


Notes: *The File Content above was truncated for space and readability, the query should allow a minimum of 0 and maximum of 100 sample ids results per request as shown in the File Sample. Dump files are the exception to this rule, these can have thousands of potential sample ID's.*

For error handling, if an error is encountered while querying the client system, the JSON Query Results file should return the following only:

```
{
                "selectedQuery":"QueryType",
                "sampleIds":[
                  "SampleId1",
                  "SampleId2",
                  "SampleId3",
                  "…",
                  "…",
                  "SampleId100"],
                "error":"Error Message Details i.e. Query Timed out"
        }
```

iDetic will continuously monitor the configured network output location for new json query result files. Once a new file is found, iDetic will pick up the new file, read its contents, delete the file, parse the results, and return the results back to the iDetic Web UI.

4.0 Report Integration

The API can also be used to integrate tube/storage information into your existing reports, without needing to use the iDetic Web UI. If you prefer to connect directly to the database without an API, please reach out and we can discuss other options.

API Specification

The iDetic Web API is meant to provide easy access to specific iDetic data using REST architecture. Our API will provide RESTful service(s) which will only work over the Web. REST is an architectural style that specifies constraints, such as the uniform interface, that if applied to a web service induce desirable properties, such as performance, scalability, and modifiability. In the REST architectural style, data and functionality are considered resources and are accessed using Uniform Resource Identifiers (URIs), typically links on the Web. The resources are acted upon by using a set of simple, well-defined operations. The REST architectural style constrains an architecture to a client/server architecture and is designed to use a stateless communication protocol, typically HTTP.

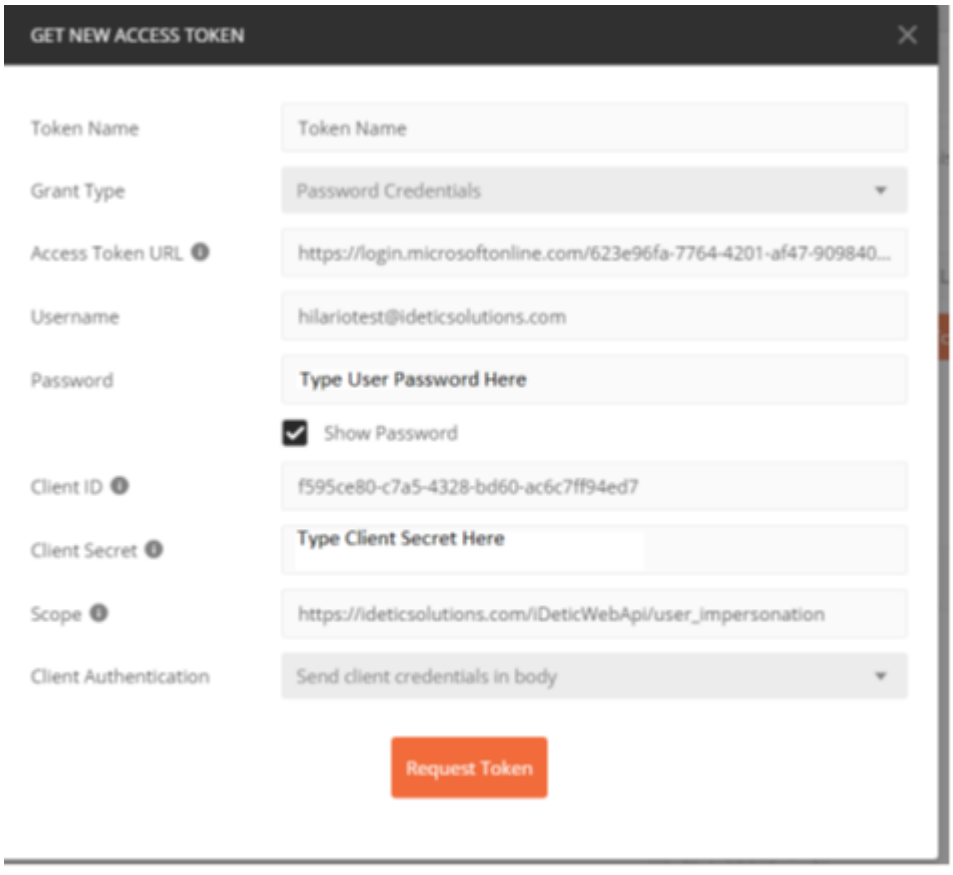
Security

The iDetic Web API is hosted in the Azure Cloud platform. It leverages Azure AD, Microsoft identity platform and the OAuth 2.0 resource owner password credential grant to provide access and security to the API. Any user's credentials can be used along with the client secret to access the API. You can find more about the Microsoft identity platform and the OAuth 2.0 resource owner password credential grant at: https://docs.microsoft.com/en-us/azure/active-directory/develop/v2-oauth-ropc

Obtaining a Token

You will need an access token in order to be able to use our secured REST web services.

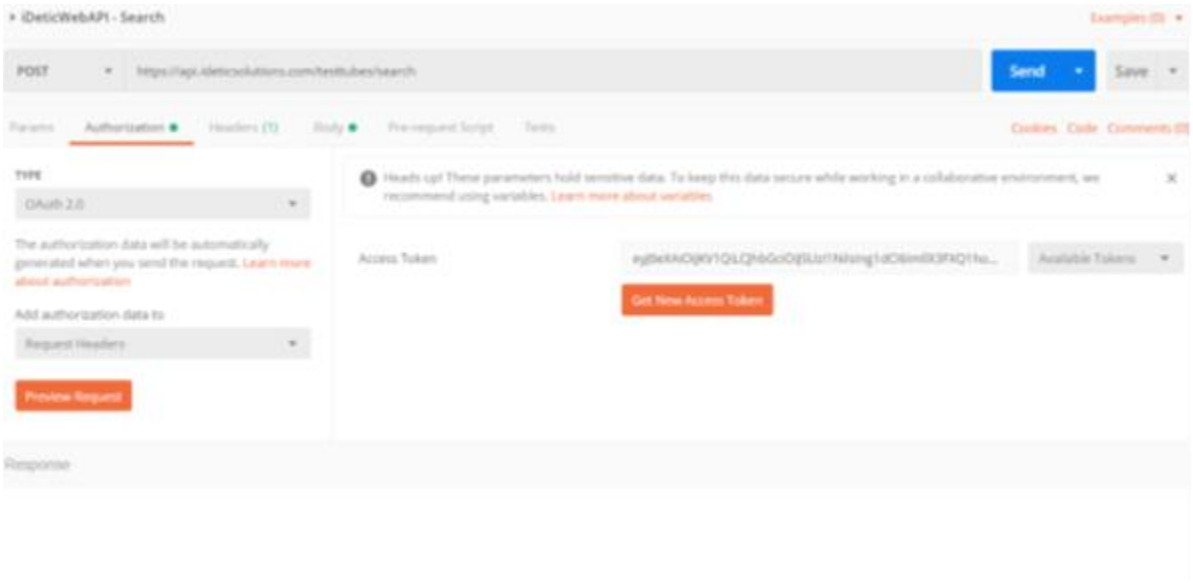
1) Access Token URL: https://login.microsoftonline.com/623e96fa-7764-4201-af47-90984061ccba/oauth2/v2.0/token
2) Client ID: f595ce80-c7a5-4328-bd60-ac6c7ff94ed7
3) Scope: https://ideticsolutions.com/iDeticWebApi/user_impersonation
4) The username and password credentials can be of any iDetic user.
5) The client secret must be provided by the iDetic team.
6) Please see the POSTMAN sample below to learn how to obtain a token.

a. 

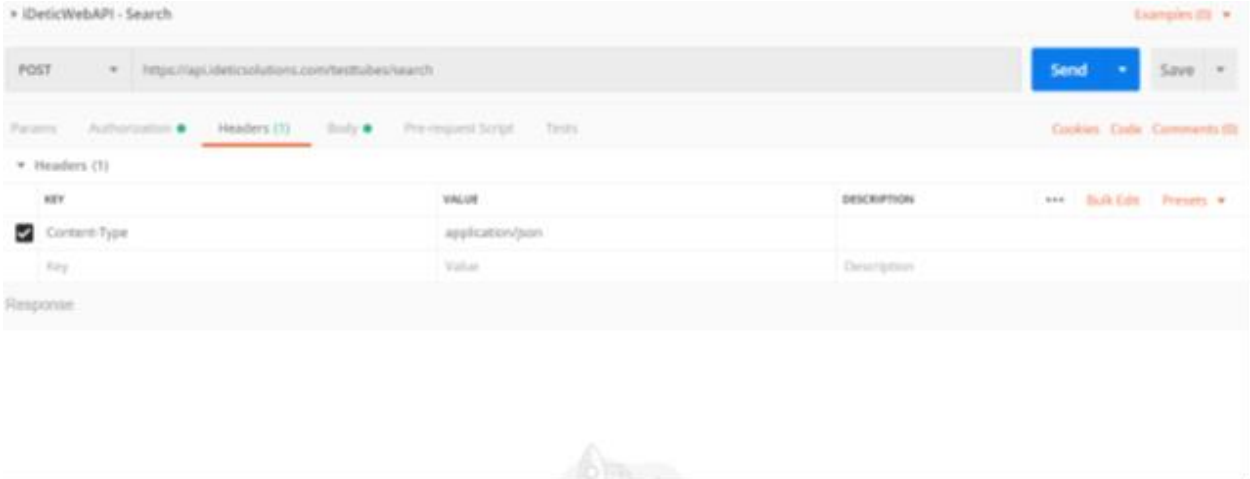
Once a token is obtained, this token can be used to make a secured Web service request. Tokens will expire so you will have to request another token after expiration.

Using the Token

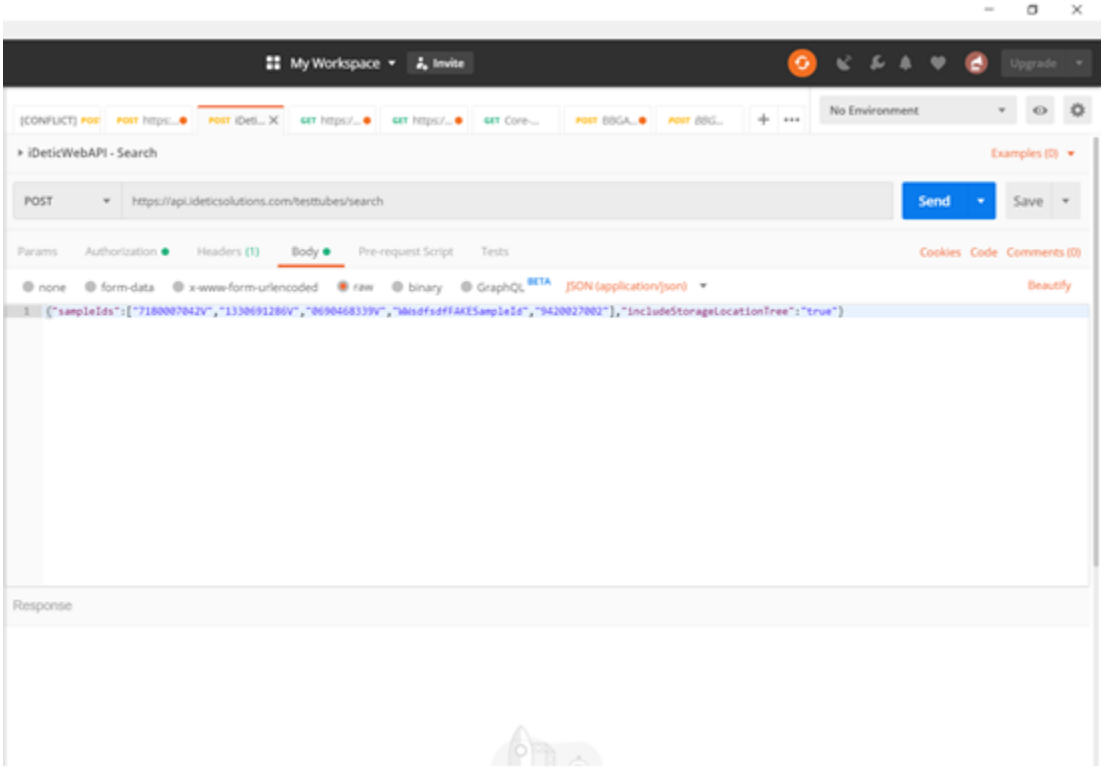
Once you obtain a token you will need to add the token to the Authorization header of the HTTP Request. Please see the POSTMAN sample below to learn how to add the Authorization header to an HTTP Request.



You will also need to set the Content-Type to application/json.



If the REST Web is of type POST or PUT it will require a body.



Available REST Web Services

Type: POST URI: https://api.ideticsolutions.com/testtubes/search

Parameters: sampleIds and includeStorageLocationTree

> If includeStorageLocationTree is true then the entire storage tree will be returned. i.e. StorageLocation ->SubStorageLocation-> SubStorageLocation

Body Sample:

> {"sampleIds":["VAL00000043","1330691286V","0690468339V","WWsdfsdfFAKESampleId","9420027002"],"includeStorageLocationTree":"true"}

Response Sample:


```
{
   "VAL00000043": [
      {
         "organization": "Validation",
         "site": "Atlanta",
         "rackId": "R-ATL0000203",
         "slotNumber": 66,
         "storageLocation": null,
         "storageLocationBarcode": null,
         "storageLocationTree": null,
         "testTubeCreatedBy": {
            "id": 2,
            "displayName": "Hilario",
            "emailAddress": "hilario@ideticsolutions.com",
            "isAccountEnabled": true,
            "firstName": "Hilario",
            "lastName": "Urquieta"
         },
         "testTubeCreatedDate": "2019-09-10T21:38:25.95",
         "testTubeLastUpdatedBy": {
            "id": 2,
            "displayName": "Hilario",
            "emailAddress": "hilario@ideticsolutions.com",
            "isAccountEnabled": true,
            "firstName": "Hilario",
            "lastName": "Urquieta"
         },
         "testTubeLastUpdateDate": "2019-09-10T21:38:25.95"
      }
   ],
   "1330691286V": [],
   "0690468339V": [],
   "WWsdfsdfFAKESampleId": [],
   "9420027002": []
}
```

Response Sample with Send-Out Data:
(App feature which shows tubes that were shipped to other locations)


```
{
   "VALsendout2": [
      {
         "organization": "Validation",
         "site": "Atlanta",
         "rackId": "SendOut",
         "slotNumber": 0,
         "storageLocation": "LabCorp",
         "storageLocationBarcode": "",
         "storageLocationTree": null,
         "testTubeCreatedBy": {
            "id": 0,
            "displayName": "Hilario Urquieta",
            "emailAddress": "hilariotest@ideticsolutions.com",
            "isAccountEnabled": null,
            "firstName": null,
            "lastName": null
         },
         "testTubeCreatedDate": "2021-03-14T11:10:20.585",
         "testTubeLastUpdatedBy": {
            "id": 0,
            "displayName": "Hilario Urquieta",
            "emailAddress": "hilariotest@ideticsolutions.com",
            "isAccountEnabled": null,
            "firstName": null,
            "lastName": null
         },
         "testTubeLastUpdateDate": "2021-03-14T11:10:20.585"
      }
   ]
}
```