

Query Tool Application Programming Interface (API)





THIS PAGE HAS BEEN LEFT INTENTIONALLY BLANK

Table of Contents

Query Tool Application Programming Interface	3
Objective	3
Authentication	4
Authentication.....	4
Study API.....	5
Get All Study Information	5
Get all studies associated with a form structure.....	7
Form Structure API.....	8
Get form structures for a study	8
Data Elements API	10
Get data elements for a form structure	10
Data API.....	12
Get data from multiple form structures without doing joins	12
Get data from multiple form structures for the given studies	14
Get data with filter and joins.....	15

QUERY TOOL APPLICATION PROGRAMMING INTERFACE

The Query Tool Application Programming Interface (API), a RESTful API that is connected to a micro service that allows users to make HTTPS requests using GET and POST. Users will be able to query data from different endpoints using Python, JavaScript, R and other tools that use Restful APIs.

OBJECTIVE

This chapter provides information for users on how to:

- ❖ To log in to the Query Tool API
- ❖ Enter parameter information for each endpoint

For information about the Query Tool API Endpoints please access the API Documentation for the BRICS Instance:

BRICS Instance	Authentication API	Query API
NEI	https://brics.nei.nih.gov/gateway/authentication/sw-agger-ui.html	https://brics.nei.nih.gov/gateway/query-api/sw-agger-ui.html
CdRns	https://cdrns.nih.gov/gateway/authentication/sw-agger-ui.html	https://cdrns.nih.gov/gateway/query-api/sw-agger-ui.html
FITBIR	https://fitbir.nih.gov/gateway/authentication/sw-agger-ui.html	https://fitbir.nih.gov/gateway/query-api/sw-agger-ui.html
PDBP	https://pdbp.ninds.nih.gov/gateway/authentication/sw-agger-ui.html	https://pdbp.ninds.nih.gov/gateway/query-api/sw-agger-ui.html

For more information and examples on the endpoints, please refer to the Jupyter Notebook.

AUTHENTICATION

In order to use the Query Tool API, the user first must POST their credentials (i.e., Username and Password) for the Authentication endpoint.

Once, the user enters his/her credentials, a token will be provided, which will be used for all subsequent endpoints. If the token expires, the user can retrieve a new token or renew it. The section on Authentication, will provide more information entering credentials and retrieving the token using HTTPS requests with Python.

AUTHENTICATION

To log in the Query Tool API the user needs to log in and retrieve the access token that is used for subsequent endpoints.

This service will authenticate a user permission to use BRICSQuery Tool API.

The following information is needed:

Endpoint URL: <https://bricsnei-stage.cit.nih.gov/gateway/authentication/user/login>

Parameters:

Headers	Response content type: text/plain' Content-Type: application/x-www-form-urlencoded
Data	Username Password
POST Response	response = requests.post("https://bricsnei-stage.cit.nih.gov/gateway/authentication/user/login", headers=headers data=Username and Password)

Figure 1: Example of input information

```
#Login in to API
url = "https://bricsnei-stage.cit.nih.gov/gateway/authentication/user/login"
headers = {
    'accept': 'text/plain',
    'Content-Type': 'application/x-www-form-urlencoded'
}

data = {'password':UserPassword,
        'username': UserUsername}

response = requests.post(url, headers=headers, data=data)
```

Output

Output Format	Text
---------------	------

Figure 2: Example of output Information

```
#Login in check
if response.status_code == 200:
    print("Login Successful")
    token=response.text
    print(f'Here is your token: {token}')
elif response.status_code != 200:
    print(response.status_code)
    print("Login not Successful. Please check username and password. If error still occurs reach out to system administrator. THIS CODE WILL NOT PROCEED")
```

Login Successful
Here is your token: eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJocm9kbmVSM5IsIm9yZyI6LTesImF1dGgiOiJST0x0eXh0RjQ1R3R05BU1ksUk9MRV9TVFVWESXSt0eXh0RjQ1R3R05BU11fRUZPUk0sUk9MRV9VU0VSLFJPT0VU1RVRV1fQURNSU4sUk9MRV9BQ08NPVU5UXBFTUI0LFJPT0VFTUUVUQVNUURZLFJPT0VFTUUVFU1ksUk9MRV9NRVRBU1RVRV1fQURNSU4sUk9MRV9ESUNUSU90QVJZXBFE1fQURNSU4iLCJmdwxsImFtZSI6IjVjZG5leSwgSGVhdGh1ciIsIm1kIjoyO0csInR1bmFudC16ImV5ZWd1bmUiLCJleHAiOjE1ODc2NzA0MTZ9.6cH-zynaBTT5K8pQZm5dUx16EDB7aqAqg5cwomJvaeGtf5Jt3tueMvq_M3Jff5QvJ6DeCAeomIWIPZgA98RBVg

STUDY API

The following endpoints will return the study profile information for a study. As mentioned in the Authentication section, a token is needed to retrieve data for all subsequent endpoints.

There are two endpoints that allow users to the do following:

1. Get study profile information for all studies.
2. Get study profile information for one study using a Study Prefix
3. Get studies that have submitted data to a form structure using the Form Structure Shortname.

Below are the endpoints for retrieving information about studies and examples.

GET ALL STUDY INFORMATION

This service will return all the studies that have data in the instance. Optional it will return information for a study with the study Prefix ID (Study ID) provided. The Study Prefix ID can be retrieved using this endpoint.

The following information is needed

Endpoint URL: <https://bricsnei-stage.cit.nih.gov/gateway/query-api/study>

Parameters:

Headers	Response content type: application/json Content-Type: application/json Authorization: Bearer + Token
Data	Optional: prefixedId =

GET Response	<pre>response = requests.get("https://bricsnei-stage.cit.nih.gov/gateway/query-api/study", headers=headers) response = requests.get("https://bricsnei-stage.cit.nih.gov/gateway/query-api/form/study?prefixedId=STUDYID", headers=headers)</pre>
---------------------	---

Figure 1: Example of Input

```
url = "https://bricsnei-stage.cit.nih.gov/gateway/query-api/study"

headers = {
    'accept': 'application/json',
    'Content-type': 'application/json',
    'Authorization': 'Bearer ' + token
}

query = requests.get(url, headers = headers)
query

<Response [200]>
```

Output

Output Format	JSON
----------------------	------

Output Description

Output Object	Required (Yes/No)
Study Title	Yes
Study ID	Yes
Abstract	Yes
Principal Investigator	Yes

Figure 2: Example of Output

```
output = query.json()
output

[{'abstract': 'The Age-Related Eye Disease Study (AREDS) is a major clinical trial sponsored by the National Eye Institute, one of the federal government's National Institutes of Health. The AREDS was designed to learn more about the natural history and risk factors of age-related macular degeneration (AMD) and cataract and to evaluate the effect of high doses of vitamin C, vitamin E, beta-carotene and zinc on the progression of AMD and cataract. Results from the AREDS showed that high levels of antioxidants and zinc significantly reduce the risk of advanced AMD and its associated vision loss. These same nutrients had no significant effect on the development or progression of cataract.',
  'status': 'Public',
  'id': 'NEI BRICS-STUDY0000205',
  'title': 'Age-Related Eye Disease Study (AREDS)',
  'pi': 'Kerry Goetz'},
 {'abstract': 'In October 2009, the FDA, the National Eye Institute (NEI), and the Department of Defense (DoD) launched the LASIK Quality of Life Collaboration Project (LQQLCP) to help better understand the potential risk of severe problems that can result from LASIK. The project aimed to develop a tool to determine the percent of patients who develop difficulties performing their usual activities following LASIK, and to identify predictors for those patients. At the time we developed our project, there was a limited amount of valid scientific data on certain patient-reported outcomes (PROs) related to LASIK. A PRO is a report of a condition experienced and reported by the patient, not the health care provider. Most LASIK studies used tools, such as questionnaires, to assess visual symptoms, but only after the surgery. The Patient-Reported Outcomes with LASIK (PROWL) disclaimer icon studies in the LQQLCP assessed visual symptoms both before and after their LASIK surgery to identify changes over time. The studies also measured the impact symptoms directly had on performing usual activities, which had not previously been done.',
  'status': 'Public',
  'id': 'EYEGENE-STUDY0000204',
  'title': 'LASIK Quality of Life Collaboration Project',
  'pi': 'Kerry E Goetz'}
```

GET ALL STUDIES ASSOCIATED WITH A FORM STRUCTURE

Returns all the studies that have data submitted to the form structure

The following is needed

Endpoint URL: <https://bricsnei-stage.cit.nih.gov/gateway/query-api/study/form?formName=>

Parameters:

Headers	Response content type: application/json Content-Type: application/json Authorization: Bearer + Token
Data	Form Structure Short Name
GET Response	response = requests.get("https://bricsnei-stage.cit.nih.gov/gateway/query-api/study/form?formName=Form Structure Shortname", headers=headers)

Figure 1: Example of Input


```
#get List of form structures

url = "https://bricsnei-stage.cit.nih.gov/gateway/query-api/study/form?formName="
header = {
  'accept': 'application/json',
  'Content-type': 'application/json',
  'Authorization': 'Bearer ' + token
}

formstructureshortname = input()

eyeGENEDemographics

query = requests.get(url + formstructureshortname,headers=header)
```

Output

Output Format	JSON
----------------------	------

Output Description

Output Object	Required (Yes/No)
Study Title	Yes
Study ID	Yes
Abstract	Yes
Principle Investigator	Yes

Figure 2: Example of output

```
formstructureinformation = query.json()
formstructureinformation

[{'form': 'eyeGENEDemographics',
  'studies': [{'abstract': 'The National Ophthalmic Disease Genotyping and Phenotyping Network (eyeGENE) is a research venture created by the National Eye Institute (NEI), part of the National Institutes of Health (NIH), in response to promising scientific discoveries in genetics. eyeGENE aims to advance studies of eye diseases and their genetic causes by giving researchers access to DNA samples, clinical information, and patients looking to participate in research studies and clinical trials. Contact Us: Phone: (301) 435-3032 Email: neieyegeneinfo@nei.nih.gov ',
    'status': 'Public',
    'id': 'EYEGENE-STUDY0000203',
    'title': 'eyeGENE',
    'pi': 'Kerry Goetz'}]}
```

FORM STRUCTURE API

The Form Structure API uses the Study Prefix ID and returns the form structures that have data submitted against in a JSON format. To learn more about retrieving the Study Prefix ID, please refer to the section on the Study API endpoints.

GET FORM STRUCTURES FOR A STUDY

Returns all the form structures that have data submitted for the study.

The following is needed

Endpoint URL: https://bricsnei-stage.cit.nih.gov/gateway/query-api/form/study?prefixedId=

Parameters:

Headers	Response content type: application/json Content-Type: application/json Authorization: Bearer + Token
Data	STUDYID
GET Response	response = requests.get("https://bricsnei-stage.cit.nih.gov/gateway/query-api/form/study?prefixedId=STUDYID", headers=headers)

Figure 1: Example of Input

```
url = "https://bricsnei-stage.cit.nih.gov/gateway/query-api/form/study?prefixedId="

headers = {
    'accept': 'application/json',
    'Content-type': 'application/json',
    'Authorization': 'Bearer ' + token
}

studyid = input("Enter Study PrefixID")

Enter Study PrefixID NEI BRICS-STUDY0000205

query = requests.get(url + studyid, headers = headers)
```

Output

Output Format	JSON
----------------------	------

Output Description

Output Object	Required (Yes/No)
Study Prefix ID	Yes
Form Structure Short Name	Yes
Form Structure Title	Yes

Figure 2: Example of Output

```
studyFormStructuredata = query.json()
studyFormStructuredata
```

```
[{'studyId': 'NEI BRICS-STUDY0000205',
  'forms': [{'id': 262,
    'shortName': 'AREDS2_ATSReRandomization',
    'title': 'AREDS2_ARR ATS Re-Randomization Clinical form',
    'version': '1.1'},
    {'id': 260,
    'shortName': 'AREDS2_AdverseEventReport',
    'title': 'AREDS2_ADV Adverse Event Report clinical data form',
    'version': '1.1'},
    {'id': 268,
    'shortName': 'AREDS2_AdverseEventReview',
    'title': 'AREDS2_AER Adverse Event Review Clinical Form',
    'version': '1.1'},
    {'id': 269,
    'shortName': 'AREDS2_CardiovasclrOutcms',
    'title': 'AREDS2_COR Cardiovascular Outcomes Study Report Clinical form',
    'version': '1.1'},
    .....
  ]
}
```

DATA ELEMENTS API

The data element API uses the form structure short name and returns the data elements within that form structure. The output will be in a JSON format and provide information about the data element such as the position in the form structure, the title and variable name.

The information about the data element is useful when filtering data in the Data API.

Below is information and examples for retrieving the data elements for that form structure.

GET DATA ELEMENTS FOR A FORM STRUCTURE

Return all data elements associated with the form structure.

The following is needed

Endpoint URL: <https://bricsnei-stage.cit.nih.gov/gateway/query-api/dataElement/form/>

Parameters:

Headers	Content-Type: application/json Authorization: Bearer + Token
Data	Form Structure Short Name

GET Response	response = requests.get("https://bricsnei-stage.cit.nih.gov/gateway/query-api/dataElement/form/Form Structure Shortname", headers=headers)
---------------------	--

Figure 1: Example of Input

```

url = "https://bricsnei-stage.cit.nih.gov/gateway/query-api/dataElement/form/"

headers = {
#   'accept': 'application/json',
  'Content-type': 'application/json',
  'Authorization': 'Bearer ' + token
}

# print("Input Form Structure Short Name")
formstructureshortname = input()

eyeGENEGenomics

dataelementapiquery = requests.get(url + formstructureshortname, headers = headers)

```

Output

Output Format	JSON
----------------------	------

Output Description

Output Object	Required (Yes/No)
Data Element Name	Yes
Form Structure Group Name	Yes
Data Element Title	Yes
Data Element Short Description	Yes
Data Element Data Type	Yes

Figure 2: Example of Output

```
dataelementapiinformation = dataelementapiquery.json()
dataelementapiinformation

[{'name': 'Main',
  'position': 0,
  'threshold': 1,
  'dataElements': [{'id': 230,
    'name': 'GUID',
    'title': 'GUID',
    'description': 'Global Unique ID (GUID) which uniquely identifies a subject',
    'dataType': 'GUID',
    'inputRestriction': 'Free-Form Entry',
    'requiredType': 'Required'},
    {'id': 222,
    'name': 'eyeGENEID',
    'title': 'eyeGENE Subject Identifier',
    'description': 'Subject identifier related to eyegene biospecimen record. Identifier can be used to request subject contact or specimen through the eyeGENE Coordinating Center',
    'dataType': 'Numeric Values',
    'minimumValue': 0.0,
    'maximumValue': 10000.0,
    'inputRestriction': 'Free-Form Entry',
    'requiredType': 'Recommended'},
    ...
  ]
}
```

DATA API

The Data API allows users to retrieve data for one or more studies and form structures. Similar to the Query Tool, users are able to (1) Download data for form structures within a study, (2) Join form structures for one or more studies, (3) Filter on data elements with the advance Boolean Search.

Please note for the current release of King Kong, associated files and images are not downloadable .

Below are the endpoints and examples for retrieving data from studies and form structures.

GET DATA FROM MULTIPLE FORM STRUCTURES WITHOUT DOING JOINS

Returns data for multiple form structures without doing joins could also include study association if required.

The following is needed

Endpoint URL: <https://bricsnei-stage.cit.nih.gov/gateway/query-api/data/bulk/form/study>

Parameters:

Headers	Content-Type: application/json Authorization: Bearer + Token
Data	Form Structure Short Name Optional: Study Prefix IDs

GET Response	response = requests.get("https://bricsnei-stage.cit.nih.gov/gateway/query-api/data/bulk/form/study", headers=headers,json=data)
---------------------	---

Figure 1: Example of Input

```

multipleformsheader = {
  'Content-type': 'application/json',
  'Authorization': 'Bearer ' + token
}

multipleformsurl = "https://bricsnei-stage.cit.nih.gov/gateway/query-api/data/bulk/form/study"

multipleformsfilter = {
  "flattened": "false",
  "formStudies": [
    {
      "form": "AREDS2_AdverseEventReview",
      "studies": [
        "NEI BRICS-STUDY0000205"
      ]
    },
    {
      "form": "PROWL_Demo_2",
      "studies": [
        "EYEGENE-STUDY0000204"
      ]
    }
  ]
},
"outputFormat": "csv"
}

multipleformsquery = requests.post(multipleformsurl,headers = multipleformsheader,json = multipleformsfilter)
multipleformsquery

<Response [200]>

```



Output

Output Format	Zip Files
----------------------	-----------

Output Description

Output Object	Required (Yes/No)
CSV File of Data	Yes

Figure 2: Example of Output

Name	Date modified	Type
 query_result_AREDS2_AdverseEventReview_2020-04-10T21-54-526056494659686228737.csv	4/10/2020 5:56 PM	Microsoft Excel C...
 query_result_PROWL_Demo_2_2020-04-10T21-54-535863629050270525402.csv	4/10/2020 5:56 PM	Microsoft Excel C...

GET DATA FROM MULTIPLE FORM STRUCTURES FOR THE GIVEN STUDIES

Returns data for multiple form structure to study associations, without doing joins

The following is needed

Endpoint URL: <https://bricsnei-stage.cit.nih.gov/gateway/query-api/data/bulk/study/form>

Parameters:

Headers	Response content type: application/zip Content-Type: application/json Authorization: Bearer + Token
Data	Form Structure Short Name Study Prefix IDs
GET Response	response = requests.get("https://bricsnei-stage.cit.nih.gov/gateway/query-api/data/bulk/form/study", headers=headers,json=data)

Figure 1: Example of Input

```

multipleformsstudy = {
  'accept': 'application/zip',
  'Content-type': 'application/json',
  'Authorization': 'Bearer ' + token
}

multipleformsstudyurl = "https://bricsnei-stage.cit.nih.gov/gateway/query-api/data/bulk/study/form"

multipleformsstudyfilter = {
  "flattened": "false",
  "outputFormat": "csv",
  "studyForms": [
    {
      "forms": [
        "eyeGENE_Clinical"
      ],
      "study": "NEI_BRICS-STUDY0000207"
    },
    {
      "forms": [
        "PROWL_Demo_2"
      ],
      "study": "EYEGENE-STUDY0000204"
    }
  ]
}

...

multipleformsstudyquery = requests.post(multipleformsstudyurl,headers = multipleformsstudy,json = multipleformsstudyfilter)
multipleformsstudyquery

<Response [200]>

```



Output

Output Format	Zip Files
----------------------	-----------

Output Description

Output Object	Required (Yes/No)
CSV File of Data	Yes

Figure 2: Example of Output

Name	Date modified	Type	Size
 query_result_eyeGENE_Clinical_2020-04-17T13-41-08613314093962330537.csv	4/17/2020 9:41 AM	Microsoft Excel C...	
 query_result_PROWL_Demo_2_2020-04-17T13-40-406263588248444904196.csv	4/17/2020 9:41 AM	Microsoft Excel C...	

GET DATA WITH FILTER AND JOINS

Returns data with filters and joins with up to five form structures. Data can be returned in two formats: text (csv) and JSON.

The following is needed

- Endpoint URL:** (1) <https://bricsnei-stage.cit.nih.gov/gateway/query-api/data/csv>
 (2) <https://bricsnei-stage.cit.nih.gov/gateway/query-api/data/json>

Parameters:

Headers	Response content type: application/csv or application/json Content-Type: application/json Authorization: Bearer + Token
Data	Filter on data elements, form structure and studyID
GET Response	response = requests.get("https://bricsnei-stage.cit.nih.gov/gateway/query-api/data/bulk/form/study, headers=headers,json=data)

Figure 1: Example of Input CSV


```
queryurl = "https://bricsnei-stage.cit.nih.gov/gateway/query-api/data/csv"

headers = {
  'accept': 'application/csv',
  'Content-type': 'application/json',
  'Authorization': 'Bearer ' + token }
```

```
genomicsfilter2 = {
  "formStudy": [
    {
      "form": "eyeGENEGenomics",
      "studies": ["EYEGENE-STUDY0000203"]
    },
    {
      "form": "eyeGENEDemographics",
      "studies": ["EYEGENE-STUDY0000203"]
    }
  ],
  "filter": [
    {
      "dataElement": "HGNCGeneSymb1",
      "form": "eyeGENEGenomics",
      "repeatableGroup": "Genomics Information",
      "operator": "OR",
      "value": [
        "ABCA4"
      ]
    },
    {
      "dataElement": "HGNCGeneSymb1",
      "form": "eyeGENEGenomics",
      "repeatableGroup": "Genomics Information",
      "operator": "AND",
      "value": [
        "PRPH2"
      ]
    },
    {
      "dataElement": "GeneVariantIndicator",
      "form": "eyeGENEGenomics",
      "repeatableGroup": "Genomics Information",
      "value": [
        "yes"
      ]
    }
  ]
}
```

Figure 2: Example of JSON Input

```
queryurl = "https://bricsnei-stage.cit.nih.gov/gateway/query-api/data/json"

headers = {
  'accept': 'application/json',
  'Content-type': 'application/json',
  'Authorization': 'Bearer ' + token }
```

```
query = requests.post(queryurl,headers=headers,json=genomicsfilter2)
query
```

<Response [200]>

Output

Output Format	Text
---------------	------

Output Description

Output Object	Required (Yes/No)
CSV Format with Data	Yes

Figure 3: Example of Output

```
dataset = query.text
texttodf = StringIO(dataset)
nei_data = pd.read_csv(texttodf, sep=",")
nei_data.head()
```

C:\Users\hearodne\AppData\Local\Continuum\anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3058: DtypeWarning: Columns (24) have mixed types. Specify dtype option on import or set low_memory=False.
interactivity=interactivity, compiler=compiler, result=result)

	GUID	eyeGENEGenomics.Study ID	eyeGENEGenomics.Dataset	eyeGENEGenomics.Main.GUID	eyeGENEGenomics.Main.eyeGENEID	eyeGENEGenomics.Main.AgeYrs	eyeGENEGenomics.
0	NEL_INVEV429FR0	203	NEL_BRICS-DATA0000591	NEL_INVEV429FR0	4327	52	
1	NEL_INVEV429FR0	203	NEL_BRICS-DATA0000591	NEL_INVEV429FR0	4327	52	
2	NEL_INVEV429FR0	203	NEL_BRICS-DATA0000591	NEL_INVEV429FR0	4327	52	
3	NEL_INVEV429FR0	203	NEL_BRICS-DATA0000591	NEL_INVEV429FR0	4327	52	
4	NEL_INVEV429FR0	203	NEL_BRICS-DATA0000591	NEL_INVEV429FR0	4327	52	

5 rows × 60 columns

```
len(nei_data)
```

Output Format	JSON
---------------	------

Output Description

Output Object	Required (Yes/No)
GUID	Yes
Form Structure Shortname	Yes
Study ID	Yes
Dataset ID	Yes
Form Structure Repeatable Group	Yes
Data Elements and Associated Data	Yes

Figure 4: Example of JSON Output of data

```

jsondata = query.json()
jsondata

[[{'guid': 'NEI_INVEV429FR0',
  'forms': [{'name': 'eyeGENEGenomicsV1.2',
    'studyId': 'EYEGENE-STUDY0000203',
    'datasetId': 'NEI_BRICS-DATA0000591',
    'repeatableGroups': [{'name': 'Main',
      'data': [[{'GUID': 'NEI_INVEV429FR0'},
        {'eyeGENEID': '4327'},
        {'AgeYrs': '52'},
        {'MedicalCondNEIEnrollTyp': 'Cone-Rod Dystrophy'},
        {'eyeGENEFamilyID': '6134'}]]}],
    {'name': 'Genomics Information',
      'data': [[{'GenTestReprtDate': '2019-11-19T00:00:00Z',
        {'CLIALabNam': 'PreventionGenetics'},
        {'CLIALabNum': '52D2065132'},
        {'EyeGENEGeneticTestLabMethdTyp': 'Direct Sequencing'},
        {'HGNCGeneSymb1': 'ABCA4'},
        {'NCBISeqGINum': ''},
        {'HGVSRefSeqAccnNum': 'NM_000350.2'},
        {'GeneExonList': '1-50'},
        {'GeneVariantIndicator': 'Yes'},
        {'HGVSSeqVarDNA': 'c.6069T>C'},
        {'HGVSSeqVarProtn': 'p.Ile2023Ile'},
        {'GenVarAllelicState': 'Homozygous'},
        {'GeneVariantInterpretTyp': 'benign'},
        {'RefGenVarID': 'dbsnp:rs1762114'},
        {'EyeGENETestReportFileInd': 'Yes'},
        {'EyeGENETestReportFileName': ''},
        {'EyeGENETestReportFile': ''}]]}],
  ..

```

APPENDIX

Attached are Sample API Scripts:

Script Name	Description
BRICSAPI_UserGuide	Provides examples of the API Endpoints
API_Case1	From the join of the two form structures, eyeGeneDemographics and eyeGeneGenomics, provide the GUIDs with one gene and one or more gene variant types.
API_Case2	The purpose of the script is to provide the list of GUIDs from the eyeGeneGenomics form structure that have one gene and one or more gene variant types
API_Case3A	The purpose of the script provides the GUIDs for a gene, but does not include a gene variant type.
API_Case3B	The purpose of the script provides the GUIDs with multiple genes and gene variant type.
API_GenesandExcludeGeneVariant	The purpose of the script provides the list of GUIDs with two genes and a gene variant type, but excludes additional gene variant type.
API_Genevarianttype	The purpose of the script is to provide the list of GUIDSs with two genes and a gene variant interpretation type
API_GUIDsnotinGenomics	The purpose of this script is to return the GUIDs that exist in the eyeGeneDemographics data, but not in the eyeGeneGenomics data.
HGNCGeneSymblandGeneIndicator	The purpose of this script is to return all GUIDs that have two genes and the gene indicator is "yes".



BRICS