



Quest Enterprise Suite Accuracy API Developer Guide

Last Updated: January 14, 2023

Table of Contents

Overview	3
QuickStart	3
How the JSON Response is Structured	5
How the JSON Response is Paginated	6
Provider Data Inspection Results.....	6
No Action Required.....	7
Enrich, Supplement, and Update.....	8
Additional Inspection Results (otherResults)	9
Completeness Inspection Results	9
Eligibility Inspection Results	10
Integrity Inspection Results	10
Standardization Inspection Results	11
Recommended Actions.....	11
Enrich	13
Investigate.....	14
Retain	14
Standardize	15
Supplement.....	16
Suppress.....	16
Update	17
Use Cases	18
Determine Changes from Previous Baseline	18
Find Providers with a Specific Value for a Data Element	19
Provider Eligibility	19
Find Providers Whose Eligibility Needs to Be Investigated	19
Find Providers Who Need to Be Suppressed	20
Find Providers Who Have Not Responded in 90 Days.....	21
Retrieve Data Only for Individuals or Organizations	22
Find Providers with a Specific Group Affiliation	22
Find Providers with Custom Identifiers (Cargo Columns)	22
Find Providers with Missing Contact Information	22

Overview

The Accuracy API enables you to programmatically determine what's right, wrong, and missing from your provider data, based on the latest information for providers obtained through multi-channel quarterly outreach and attestation services. For example, using the API you can:

- Identify and address high-risk challenges, such as invalid NPIs, OIG exclusions, and excessive specialties or addresses.
- Identify providers that will most impact the adequacy and health of your network.
- Identify the most critical errors by location.

Maintaining the accuracy of your provider directory with the highest fidelity information possible ensures your network's integrity, reduces regulatory risk, improves network performance, and maximizes member satisfaction.

QuickStart

This QuickStart explains how to:

- Obtain an authorization token so you can access the Accuracy API endpoints.
- Get the accuracy baseline for all providers for a specific project.
- Identify the actions recommended to ensure the accuracy of provider data.

1. Obtain an authorization token:

- a. Request a client id and client secret from Quest Analytics Client Services.
- b. Submit a request to the auth endpoint, passing the client id and client secret in the request body. For example:

POST /token HTTP/1.1

Host: https://uat-api.questanalytics.com/enterprise/auth

Content-Type: application/json

Subscription-Key: {{subscriptionKey}}

Content-Length: 145

```
{
  "client_id": "{{clientId}}",
  "client_secret": "{{clientSecret}}",
  "scope": "default",
  "grant_type": "client_credentials"
}
```

The POST request returns a large authorization token (approximately 1800 characters).

- ### 2. GET the accuracy baseline (all data for all providers) for a specific project, passing the ProjectId in the URL and the authorization token in the request body. For example:

GET /enterprise/accuracy/v1/Inspections?projectId=123456 HTTP/1.1

Host: <https://uat-api.questanalytics.com>

Authorization: Bearer {{accessToken}}

The GET request returns a data array which contains one record for each provider. Each provider record contains three types of data:

- Data inspections for each data element.
- Inspection results for completeness, eligibility, integrity, standardization, and additional inspection results
- Recommended actions.

The recommendedActions section lists the data elements you should enrich, investigate, retain, standardize, supplement, suppress or update. For example, the following code snippet indicates that you can enrich FrequencyAtLocation, supplement ProviderLanguages, and update MiddleName:

```
"recommendedActions": {  
  "enrichReasons": [  
    "FrequencyAtLocation"  
  ],  
  "investigateReasons": [],  
  "retainReasons": [],  
  "standardizeReasons": [],  
  "supplementReasons": [  
    "ProviderLanguages"  
  ],  
  "suppressReasons": [],  
  "updateReasons": [  
    "MiddleName"  
  ]  
}
```

3. For all recommended actions except suppress, if your business logic dictates that you perform the recommend action, you can find the verified value(s) for a data element in that data element's verifiedValues. For example, the following providerLanguages data element indicates that although the client did not specify a language, data inspection has determined that English is spoken:

```
providerLanguages": {  
  "sourceValue": "",  
  "standardizedValue": null,  
  "verifiedValues": [  
    {  
      "value": "English",
```

```

        "outcome": "Enriched",
        "source": "BdxTruth",
        "date": "2022-07-26T14:05:16.791Z"
    }
]

```

Every data element in the provider record is specified with the following name-value pairs:

- "sourceValue": One or more values for the data element submitted by the client.
 - "standardizedValue": Has a value only when sourceValue does not conform to standard spelling, punctuation, or abbreviation conventions. For example, 10 Main Street would be standardized to 10 Main St.
 - "verifiedValues": An array of one or more valid, invalid, or enriched values:
 - When "outcome" is Valid, you should not change the client-supplied "sourceValue".
 - When "outcome" is Invalid, you should remove "value" from the client-supplied "sourceValue".
 - When "outcome" is Enriched, you should add "value" to the client-supplied "sourceValue".
 - "source": The attestation source used to verify the value for the data element, for example NPPES, FSMB, and ABMS.
 - "date": the date the value was attested.
4. When the recommended action is Suppress, you should remove the provider from your directory. You can find the recommended action reason in otherData, which contains additional inspection results. For example:

```

{
    "group": "Eligibility",
    "name": "Deactivated NPI",
    "outcome": "Inaccurate",
    "justification": null,
    "recommendedAction": "Suppress",
    "recommendedActionReasons": [
        "NPIDeactivated"
    ]
}

```

The recommended action reason for suppressing a provider is typically one of the following: NPI Deactivated, Medicare Opt-out, Medicare Precluded, or OIG Excluded.

How the JSON Response is Structured

The JSON response returns the following information:

- Pagination information
- One record for each Provider, containing:
 - The data inspection results for each data element (name, address, phone, and so on)
 - Additional inspection results (completeness, eligibility, integrity, and standardization)
 - Recommended actions

How the JSON Response is Paginated

The first part of the JSON response indicates how the response is paginated, based on the URL parameters passed in the request. For example:

```
"pageNumber": 1,
"pageSize": 1000,
"firstPage": "/v1/Inspections?projectId=47145&pageNumber=1&pageSize=1000",
"lastPage": "/v1/Inspections?projectId=47145&pageNumber=2&pageSize=1000",
"totalPages": 2,
"totalRecords": 1013,
"nextPage": "/v1/Inspections?projectId=47145&pageNumber=2&pageSize=1000",
"previousPage": null,
```

In the preceding example, the JSON indicates that the response is paginated as follows:

- `pageNumber`: This is the first page of data in the response.
- `pageSize`: The response has 1000 providers on each page.
- `firstPage`: The URL from which to retrieve the first page of data.
- `lastPage`: The URL from which to retrieve the last page of data.
- `totalPages`: The response has 2 pages of data.
- `totalRecords`: The response has the data for 1013 providers.
- `nextPage`: The URL from which to retrieve the next page of data. In this example, this is the same as the URL as `lastPage` because there are only two pages.
- `previousPage`: The URL from which to retrieve the previous page of data. In this example, this is null because the current page is the first page.

Note: You typically use `pageNumber` and `pageSize` as query string parameters in subsequent API requests to iterate over the results until `nextPage` is null, indicating that you have reached the end of the results.

Provider Data Inspection Results

The data for each provider is contained in a large JSON array named `data`, beginning just after the pagination information. There is one record in the array for each unique combination of provider and location (if a provider practices at three locations, there are three records for that provider).

The first part of the record contains the data inspection results for each data element: NPI, First Name, Location Phone, and so on. Each data element specifies one or more client-provided values (sourceValue), one or more data inspector discovered values (verifiedValues), and, where applicable, a standardized value.

The action to take on a data element depends on what values the client provided, if any, and what values the data inspectors found, if any.

No Action Required:

- No client-provided values and no inspector-discovered values.
- Client-provided values exist but there are no inspector-discovered values.
- Client-provided values and inspector-discovered values match.

Action Required:

- Supplement (add values): No client-provided values and verified inspector-discovered values.
- Enrich (add values): Unverified client-provided values and verified inspector-discovered values.
- Update (replace values): Invalid client-provided values and verified inspector-discovered values.

No Action Required

The three scenarios in which no action is required are indicated by null values for sourceValue and/or verifiedValues, or by matching values for sourceValue and verifiedValues.

If there are no client-provided values and no inspector-discovered values, then the entire data element is null. For example:

```
"locationPhone": null
```

If client-provided values exist but data inspectors could not verify those values, then sourceValue is populated and verifiedValues is null. For example:

```
"locationPhone": {  
  "sourceValue": "5104736707",  
  "standardizedValue": null,  
  "verifiedValues": []  
}
```

If the client-provided values and the inspector-discovered values match, then sourceValue and verifiedValues match and “Valid” is returned in outcome. For example:

```
"locationPhone": {  
  "sourceValue": "5107521000",  
  "standardizedValue": "5107521000",  
  "verifiedValues": [  
    {  
      "value": "5107521000",  
      "outcome": "Valid",
```

```

        "source": "NppesTruth",
        "date": "2022-11-01T00:00:00Z"
    }
]
}

```

Enrich, Supplement, and Update

Consider the following scenario for the providerLanguages element:

- Client submits English, French, and Mandarin Chinese.
- Inspection results indicate that:
 - English and French are valid, and no action is required (keep sourceValue)
 - Mandarin is invalid and needs to be removed (update sourceValue)
 - The Provider also speaks Portuguese and Russian, which need to be added (enrich sourceValue)

Given this scenario, the providerLanguages data element returned by the API would look like this:

```

"providerLanguages": {
  "sourceValue": "English;French;Mandarin Chinese",
  "standardizedValue": null,
  "verifiedValues": [
    {
      "value": "English",
      "outcome": "Valid",
      "source": "BdxTruth",
      "date": "2022-11-29T19:25:15.7Z"
    },
    {
      "value": "French",
      "outcome": "Valid",
      "source": "BdxTruth",
      "date": "2022-11-29T19:25:15.7Z"
    },
    {
      "value": "Portuguese",
      "outcome": "Enriched",
      "source": "BdxTruth",
      "date": "2022-11-29T19:25:15.7Z"
    }
  ]
}

```

```

{
  "value": "Russian",
  "outcome": "Enriched",
  "source": "BdxTruth",
  "date": "2022-11-29T19:25:15.7Z"
},
{
  "value": "Mandarin Chinese",
  "outcome": "Invalid",
  "source": "BdxTruth",
  "date": "2022-11-29T19:25:15.7Z"
}
]
}

```

QES evaluates the validity of the client-provided source values only for the following data elements: NPI, Entity Type, Name, Location Name, Ord NPI, Address, Specialty, Phone, Fax, Panel Status (PS), Gender, Office Hours, Languages, Affiliations, Email, Website, Tin, and License.

QES does NOT evaluate the validity of client-provided source values for: ADA Features, Frequency at Location, Bboard certs, Medicaid License, Race, Ethnicity, Telehealth, Public Transport, Cultural Competency, Taxonomy, or Areas of Expertise. The reason they are not validated is because many of these values are not used by clients or are not stored in an industry-standard way. Therefore, when the recommended action is to enrich one of these data elements, the recommendation is made without regard for whether the client's own values for these data elements is valid or invalid. The best practice is for clients to make their own determination on whether the enriched values for some of these data elements are useful.

Additional Inspection Results (otherResults)

The otherResults portion of the record for each provider contains additional inspection results for completeness, eligibility, integrity, and standardization.

Completeness Inspection Results

The Completeness inspection results indicate whether a value is Accurate (present) or Inaccurate (absent) for the following values in the data array: NPI, Name, Address, Location Phone, Location Fax, and Email. In other words, did the Provider submit data for these six data elements? The recommendedAction, recommendedActionReasons, and justification are always null. If a data element is

missing, the results of other inspectors for that data element will indicate whether you should enrich your network with new valid values.

Eligibility Inspection Results

The Eligibility inspection results indicate whether the provider is eligible to be practicing. Accurate indicates the Provider passed the Eligibility inspection and is eligible. Inaccurate indicates the Provider failed the Eligibility inspection. In the event the Provider fails the Eligibility inspection, the recommended action is almost always Suppress, because you cannot include a provider in the network if they are not eligible:

Inspection Result	Description	Recommended Action
NPI Deactivated	The Provider's National Provider Identifier is no longer active.	Suppress
Medicare Opt-out	The Provider opted out of Medicare.	Suppress
Medicare Precluded	Medicare removed doctor from approved list.	Suppress
OIG Excluded	The Office of Inspector General removed doctor from approved list.	Suppress
Provider License	The Provider's license has expired.	Investigate
Provider Retired	The Provider is no longer practicing.	Investigate

When the provider's license has expired or the provider is no longer practicing, the recommended action is to Investigate, because it is possible that the provider will renew the expired license or will resume practicing. If investigation reveals that the provider's license will never be renewed or that the provider will never return to practice, then you should remove the provider from the network (suppress).

Integrity Inspection Results

The Integrity inspection results indicate if the address or specialty data submitted by the Provider is suspect or likely incorrect, due to the quantity or combination of values submitted. The recommended action is always Investigate, because it is possible that the address and specialty information is correct.

Inspection Result	Description	Recommended Action
Excess Addresses	There are an excessive number of addresses listed for this Provider.	Investigate
Address Proximity	The combination of addresses listed for this provider are very far apart.	Investigate
Excess Specialties	There are an excessive number of specialties listed for this Provider	Investigate

Standardization Inspection Results

The Standardization inspection results indicate whether provider names, addresses, phone numbers, and fax numbers are in a consistent format, ultimately making directories easier to interpret by consumers. The recommended action is always Standardize:

Inspection Result	Examples	Recommended Action
Name not standard	Lloyd-Jones vs. Lloyd Jones	Standardize
Address not standard	Suite vs. Ste Floor vs. Fl Avenue vs. Ave	Standardize
Location Phone and/or Location Fax not standard	800-555-1212 vs. 8005551212 (800) 555-1212 vs. 8005551212	Standardize

Recommended Actions

The recommendedActions for a provider summarizes all the actions that QES recommends, based on the inspection results. There are seven recommended actions:

Suppress	The client should remove this provider. Most Eligibility Group inspectors have a result of Suppress (two have a result of Investigate)
Investigate	The data element is likely to need an update or a suppression, but the client should verify the information independently before doing so. Integrity Group inspectors have a result of Investigate.
Update	A data element is inaccurate and requires an update to the client's existing data using verified data discovered by the inspector. Validity Group inspectors that find the source provider data to be inaccurate have a result of Update.
Supplement	The provider data element is missing a verifiable value and Quest can provide that value. Validity Group inspectors with verified data that find the source provider data to be missing have a result of Supplement.
Enrich	Additional verified data can be added to the data element (the client's own data for these data elements is not being validated). Enrichment Group inspectors with verified data have a result of Enrich.
Standardize	A data element's punctuation, spelling, and abbreviations can be standardized for outreach. A standardized value changes the format of the value only; it does not indicate that the value is valid or invalid.
Retain	Keep the client-provided value. Retain does not inherently imply validity unless a verification for all inspectors has taken place.

The following JSON code snippet illustrates the structure of the recommendedActions object for a particular Provider when Quest recommends all seven actions for that provider. There is one object of the form <Action>Reasons for each recommended action: enrichReasons, investigateReasons, and so on. The values in each <Action>Reasons object are the names of the data elements that the action

applies to. For example, the investigateReasons object in the following example indicates that you should investigate the SpecialtyCodes data element.

```
"recommendedActions": {  
    "enrichReasons": [  
        "ADAFeatures",  
        "BoardCertification",  
        "Ethnicity",  
        "ProviderTelehealth",  
        "Race",  
        "Taxonomy"  
    ],  
    "investigateReasons": [  
        "SpecialtyCodes"  
    ],  
    "retainReasons": [  
        "EntityType",  
        "Gender",  
        "LocationPhone",  
        "NPI",  
        "PracticingAtOffice",  
        "ProviderLanguages",  
        "ProviderLicenses",  
        "TIN"  
    ],  
    "standardizeReasons": [  
        "Address1"  
    ],  
    "supplementReasons": [  
        "OrganizationWebsite"  
    ],  
    "suppressReasons": [  
        "NPIDeactivated"  
    ],  
    "updateReasons": [  
        "GroupAffiliation",
```

```

        "HospitalAffiliation",
        "LocationFax",
        "LocationOfficeHours",
        "Prefix"
    ]
}

```

Enrich

The `enrichReasons` object within `recommendedActions` indicates the data elements you should consider enriching. Enrich means that Quest has additional verified data that you can add to the provider data elements (whether your own data for these data elements is valid).

For example, the following `enrichReasons` indicate that you should consider enriching the data elements for ADA Features, Board Certification, Ethnicity, Provider Telehealth, Race and Taxonomy:

```

"enrichReasons": [
    "ADAFeatures",
    "BoardCertification",
    "Ethnicity",
    "ProviderTelehealth",
    "Race",
    "Taxonomy"
]

```

The enriched value can be found by inspecting the corresponding data element for each `enrichReason` listed. For example consider the first `enrichReason` listed, `ADAFeatures`. To find the enriched values for ADA Features, you need to inspect the `adaFeatures` data element for the Provider:

```

"adaFeatures": {
    "sourceValue": null,
    "standardizedValue": null,
    "verifiedValues": [
        {
            "value": "Parking;Exterior_Building;Interior_Building;
Waiting_Or_Reception;Restroom;Exam_Room",
            "outcome": "Enriched",
            "source": "BdxTruth",
            "date": "2022-07-15T18:26:30.672Z"
        }
    ]
}

```

In this example, Quest is recommending that you enrich your data with the following ADA features: Parking, Exterior_Building, Interior_Building, Waiting_Or_Reception, Restroom, and Exam_Room.

Investigate

The investigateReasons object within recommendedActions indicates the data elements you should consider investigating. Investigate means that the provider data element is likely to need an update or a suppression, but that you should verify the information independently before doing so.

For example, the following investigateReasons object indicates you should consider investigating the Provider's Specialty Codes:

```
"investigateReasons": [  
  "SpecialtyCodes"  
]
```

An examination of the corresponding specialtyCodes data element for the Provider reveals why the recommended action is Investigate: Quest determined that the value is Invalid, based on data found at BdxGTruth. For example:

```
"specialtyCodes": {  
  "sourceValue": "207W00000X",  
  "standardizedValue": null,  
  "verifiedValues": [  
    {  
      "value": "207W00000X",  
      "outcome": "Invalid",  
      "source": "BdxTruth",  
      "date": "2022-07-15T18:26:30.672Z"  
    }  
  ]  
}
```

Retain

The retainReasons object within recommendedActions lists the data elements that you can leave as is. For example, the following retainReasons object indicates that you do not need to change Entity Type, Gender, Location Phone, NPI, Practicing at Office, Provider Languages, Provider Licenses, or TIN for the Provider:

```
"retainReasons": [  
  "EntityType",  
  "Gender",  
  "LocationPhone",  
  "NPI",  
  "PracticingAtOffice",
```

```

        "ProviderLanguages",
        "ProviderLicenses",
        "TIN"
    ]

```

Quest recommends that you retain data elements without change when it determines that the data elements are valid. For example, consider `EntityType`. Examining the `entityType` element for the Provider reveals that the outcome of the inspection is `Valid`, based on data found at `NppesTruth`:

```

entityType": {
    "sourceValue": "Individual",
    "standardizedValue": null,
    "verifiedValues": [
        {
            "value": "Individual",
            "outcome": "Valid",
            "source": "NppesTruth",
            "date": "2022-08-22T18:10:56.872Z"
        }
    ]
}

```

Standardize

Standardize means that Quest has determined that spelling, punctuation, or abbreviations can be standardized for outreach. For example, the following `standardizeReasons` object indicates that you can standardize Address 1 for the Provider:

```

"standardizeReasons": [
    "Address1"
]

```

This means that the `address1` data element for the Provider includes a standardized address. For example:

```

"address1": {
    "sourceValue": "1113 VILLA LINDE",
    "standardizedValue": "1113 Villa Linde Ct",
    "verifiedValues": [
        {
            "value": "1113 Villa Linde Ct",
            "outcome": "Valid",
            "source": "BdxTruth",

```

```

        date": "2022-10-06T12:57:27.039Z"
      }
    ]
  }

```

Supplement

Supplement means that the provider data element is missing a verifiable value and Quest can provide that value.

For example, the following supplementReasons object indicates that you can supplement organizationWebsite for the Provider:

```

"supplementReasons": [
  "OrganizationWebsite"
]

```

To find the new URL, you examine the organizationWebsite data element for the Provider. For example:

```

"organizationWebsite": {
  "sourceValue": null,
  "standardizedValue": null,
  "verifiedValues": [
    {
      "value": "www.eyelidpros.com",
      "outcome": "Enriched",
      "source": "BdxTruth",
      "date": "2022-07-15T18:26:30.672Z"
    }
  ]
}

```

Suppress

Suppress indicates that Quest has confidently determined that you should remove a provider.

For example, the following suppressReasons object indicates that you should remove the provider because their NPI is deactivated:

```

"suppressReasons": [
  "NPIDeactivated"
]

```

Unlike all the other recommended actions, suppress does not mean that you need to change the value of a data element within a provider record. Instead, you need to remove the Provider from your directory.

The recommendation to suppress is based on the result of the Eligibility Inspection, which are included in the otherResults object. For example:

```
{
  "group": "Eligibility",
    "name": "Deactivated NPI",
    "outcome": "Inaccurate",
    "justification": null,
    "recommendedAction": "Suppress",
    "recommendedActionReasons": [
      "NPIDeactivated"
    ]
}
```

Update

Update indicates Quest has determined that a data element is inaccurate and requires an update to your existing data using the Quest verified data.

For example, the following updateReasons object indicates that you can update a Provider's Group Affiliation, Hospital Affiliation, Location Fax, Location Office Hours, and Prefix:

```
    "updateReasons": [
      "GroupAffiliation",
        "HospitalAffiliation",
        "LocationFax",
        "LocationOfficeHours",
        "Prefix"
    ]
```

Consider the first data element listed in updateReasons, GroupAffiliation. To find the updated value for Group Affiliation, you need to inspect the verifiedValues object in the groupAffiliation data element for the Provider:

```
"groupAffiliation": {
  "sourceValue": "Jackie Smith MD",
  "standardizedValue": null,
  "verifiedValues": [
    {
      "value": "Consultants Plastic Surgery",
      "outcome": "Enriched",
      "source": "BdxTruth",
      "date": "2022-07-15T18:26:30.672Z"
    }
  ]
}
```

```

    }
  ]
}

```

Use Cases

You typically use the Accuracy API to segment and stratify provider data to focus only on providers of interest. This involves using query string parameters to limit the data returned by the API, or post-processing the full dataset for all providers where there is no combination of query string parameters that will filter the data for your particular use case.

Determine Changes from Previous Baseline

One of the most common use cases is the determine what's changed from a previous baseline:

1. Retrieve a previous baseline for a specific project by specifying the projectId and the projectRunConfigurationId. For example:

```

GET
/enterprise/accuracy/v1/Inspections?projectId=12345&projectRunConfigurationId=
123456 HTTP/1.1

```

2. Retrieve the current baseline for the same project by specifying the project Id. For example:

```

GET /enterprise/accuracy/v1/Inspections?projectId=12345 HTTP/1.1

```

3. Programmatically determine the diff between the previous baseline and current baseline using any tool of your choice. Particularly useful is to identify data elements where the verified values date has changed between the previous and the current baseline. For example, in the following JSON code snippet taken from the previous baseline, verifiedValues for providerLanguages is null:

```

providerLanguages": {
  "sourceValue": "",
  "standardizedValue": null,
  "verifiedValues": []
}

```

Whereas in the second snippet, verifiedValues has a value of English and an attestation date of January 8, 2023:

```

providerLanguages": {
  "sourceValue": "",
  "standardizedValue": null,
  "verifiedValues": [
    {
      "value": "English",
      "outcome": "Enriched",

```

```

        "source": "BdxTruth",
        "date": "2023-01-08T14:05:16.791Z"
    }
]
}

```

Find Providers with a Specific Value for a Data Element

You can get all providers with a specific value for a data element by specifying the data element and value as a query string parameter. For example, to retrieve all providers whose last name is Smith:

```
GET /enterprise/accuracy/v1/Inspections?projectId=123456&LastName=Smith
```

The complete list of query string parameters for retrieving specific data elements is as follows:

- NPI
- OrganizationNPI
- TIN
- EntityType
- Prefix, FirstName, MiddleName, LastName, Suffix, Gender, Race, Ethnicity
- Address1, Address2, City, State, Zip
- LocationName, LocationPhone, LocationFax, LocationEmail, LocationOfficeHours, FrequencyAtLocation, LocationLanguages, LocationTelehealth
- AcceptingNewPatients, NotAcceptingPatients, AcceptingNewPatientsWithLimitations, AcceptingExistingPatientsOnly
- AdaFeatures
- ProviderLanguages, ProviderLicenses, ProviderTelehealth
- BoardCertifications, MedicaidLicenses, CulturalCompetencyTrainings, Taxonomy, AreasOfExpertise, SpecialtyCode, Credentials, HospitalAffiliation, IpaAffiliation
- OrganizationWebsite
- PublicTransportations

Provider Eligibility

To determine whether a provider is eligible to practice, you submit two types of requests to the API:

- Find all providers where the recommended action is to investigate because the provider is retired, or their license is expired.
- Find all providers where the recommended action is to suppress because the NPI is deactivated.

Find Providers Whose Eligibility Needs to Be Investigated

The API results for eligibility direct you to investigate a provider, rather than suppress, when a provider's license is expired or the provider is retired, because it is possible that the provider will renew the expired license or will resume practicing. If investigation reveals the provider will not renew or resume, then you should suppress the provider (remove the provider from the network)

To get all retired practitioners:

```
GET /enterprise/accuracy/v1/Inspections?projectId=123456&InvestigateReasons=Retired
```

The results of the Eligibility Inspection for a retired provider look like this:

```
{
  "group": "Eligibility",
  "name": "Provider Retired",
  "outcome": "Inaccurate",
  "justification": null,
  "recommendedAction": "Investigate",
  "recommendedActionReasons": [
    "Retired"
  ]
}
```

Note: Each of the inspectors specifies a value for "outcome" of either "Accurate" or "Inaccurate," but the meaning varies from inspector to inspector. In this context, Accurate means the provider is still practicing and Inaccurate means the provider is Retired.

If a provider needs to be investigated because of retirement, the investigateReasons returned for that provider in recommendedActions indicates "retired":

```
"recommendedActions": {
  "enrichReasons": [],
  "investigateReasons": [
    "Retired"
  ],
  "retainReasons": [],
  "standardizeReasons": [],
  "supplementReasons": [],
  "suppressReasons": [],
  "updateReasons": []
}
```

In the case of an expired license, investigateReasons indicates "ProviderLicense."

Find Providers Who Need to Be Suppressed

To find providers who need to be suppressed, get all providers who have a deactivated NPI:

```
GET
/enterprise/accuracy/v1/Inspections?projectId=123456&SuppressReasons=NPIDeactivated
```

The results of the Eligibility Inspection for a provider whose NPI is deactivated look like this:

```
{
  "group": "Eligibility",
    "name": "NPI Deactivated",
    "outcome": "Inaccurate",
    "justification": null,
    "recommendedAction": "Suppress",
    "recommendedActionReasons": [
      "NPIDeactivated"
    ]
}
```

An outcome of Inaccurate in this context means that the NPI is deactivated. Accurate means that the NPI is still active.

If a provider needs to be suppressed, the suppressReasons returned for that provider in recommendedActions are as follows:

```
"recommendedActions": {
  "enrichReasons": [],
  "investigateReasons": [],
  "retainReasons": [],
  "standardizeReasons": [],
  "supplementReasons": [],
  "suppressReasons": [
    " NPIDeactivated "
  ],
  "updateReasons": []
}
```

Find Providers Who Have Not Responded in 90 Days

GET all data for all providers for a specific project. For example:

```
GET /enterprise/accuracy/v1/Inspections?projectId=123456
```

Search the response for all address1 elements that have a verified values date older than 90 days. For example, assuming that today's date is 2023-01-15, the Provider practicing at 6460 Pentz Rd has not provided a new value for address1 in the past 90 days because the last verified value for address1 is 2022-08-04:

```
"address1": {
  "sourceValue": "6460 PENTZ ROAD",
  "standardizedValue": "6460 Pentz Rd",
```

```

    "verifiedValues": [
      {
        "value": "6460 Pentz Rd",
        "outcome": "Valid",
        "source": "NppesTruth",
        "date": "2022-08-04T22:30:41.636Z"
      }
    ]
  }
}

```

Retrieve Data Only for Individuals or Organizations

To get data only for individuals or organizations, filter by `EntityType`:

To GET data for all individuals:

```
GET /enterprise/accuracy/v1/Inspections?projectId=123456&EntityType=Individual
```

To GET data for all organizations:

```
GET /enterprise/accuracy/v1/Inspections?projectId=123456&EntityType=Organization
```

Find Providers with a Specific Group Affiliation

To get all providers who are affiliated with a specific group, filter by `GroupAffiliation`. For example, to retrieve all providers with the ABC Medical group affiliation:

```
GET /enterprise/accuracy/v1/Inspections?projectId=1234&GroupAffiliation=ABC%20Medical
```

Note: Spaces in query string parameters must be escaped.

Find Providers with Custom Identifiers (Cargo Columns)

To get all providers with a client specific custom identifier, filter by `CargoColumns`. For example, to retrieve all providers who have a `MyCustomID` cargo column value of 1234:

```
GET /enterprise/accuracy/v1/Inspections?projectId=1234&CargoColumns.MyCustomID=1234
```

Find Providers with Missing Contact Information

To get providers with missing contact information:

1. GET all data for all providers for a specific project. For example:

```
GET /enterprise/accuracy/v1/Inspections?projectId=123456
```

5. Find all providers with a null value in the `sourceValue` or the `verifiedValues` fields for any of the address components. For example, the following contact information is missing `sourceValue` for `address1`, missing `verifiedValues` for `address2`, and missing both a `sourceValue` and `verifiedValues` for zip code:

```

"address1": {
  "sourceValue": null,

```

```

    "standardizedValue": "114 Mission Ranch Blvd",
    "verifiedValues": [
      {
        "value": "114 Mission Ranch Blvd",
        "outcome": "Valid",
        "source": "BdxTruth",
        "date": "2022-01-27T19:49:52.131Z"
      }
    ]
  },
  "address2": {
    "sourceValue": "SUITE 50",
    "standardizedValue": "Ste 50",
    "verifiedValues": []
  },
  "city": {
    "sourceValue": "CHICO",
    "standardizedValue": "Chico",
    "verifiedValues": [
      {
        "value": "Chico",
        "outcome": "Valid",
        "source": "BdxTruth",
        "date": "2022-01-27T19:49:52.131Z"
      }
    ]
  },
  "state": {
    "sourceValue": "CA",
    "standardizedValue": "CA",
    "verifiedValues": [
      {
        "value": "CA",
        "outcome": "Valid",
        "source": "BdxTruth",

```

```
        "date": "2022-01-27T19:49:52.131Z"
      }
    ]
  },
  "zip": {
    "sourceValue": null,
    "standardizedValue": null,
    "verifiedValues": []
  }
}
```