NEPOOL GIS External Interface Specification

Version 2.4

A close-up of a logo

Description automatically generated with medium confidence

Table of Contents

[1. Revision History 5](#_Toc78984945)

[2. Summary 6](#_Toc78984946)

[3. Creating an API Login 6](#_Toc78984947)

[3.1 API Login Creation Interface 6](#_Toc78984948)

[4. General API Information 7](#_Toc78984949)

[4.1 Authentication 7](#_Toc78984950)

[4.1.1 Headers 7](#_Toc78984951)

[4.1.2 Parameters 7](#_Toc78984952)

[4.1.3 Results 7](#_Toc78984953)

[4.1.4 Status Codes 8](#_Toc78984954)

[4.2 Get / Post Security 8](#_Toc78984955)

[4.2.1 Headers 8](#_Toc78984956)

[4.3 Acceptable Use 8](#_Toc78984957)

[4.4 Auxiliary/Reference Data 8](#_Toc78984958)

[4.5 Automated Emails 8](#_Toc78984959)

[4.6 Event Logging 9](#_Toc78984960)

[4.7 User Acceptance Testing (UAT) and Production Base URLs 9](#_Toc78984961)

[5. Swagger Technical API Specification 9](#_Toc78984962)

[5.1 View GIS APIs on Swagger 9](#_Toc78984963)

[5.2 Generate Client for GIS APIs 10](#_Toc78984964)

[5.3 Generate Server for GIS APIs 10](#_Toc78984965)

[6. Get APIs 10](#_Toc78984966)

[6.1 General GET API Behavior 10](#_Toc78984967)

[6.1.1 Response Codes 10](#_Toc78984968)

[6.1.2 Error Results 10](#_Toc78984969)

[6.2 Get Transferable Positions (api/Position) 11](#_Toc78984970)

[6.2.1 Parameters 11](#_Toc78984971)

[6.2.2 Success Results 11](#_Toc78984972)

[6.2.3 Error Results 11](#_Toc78984973)

[6.3 Get Pending Transfers (api/Transfers) 12](#_Toc78984974)

[6.3.1 Parameters 12](#_Toc78984975)

[6.3.2 Success Results 12](#_Toc78984976)

[6.3.3 Error Results 13](#_Toc78984977)

[6.4 Get Generator Information (api/Generators) 13](#_Toc78984978)

[6.4.1 Parameters 13](#_Toc78984979)

[6.4.2 Success Results 13](#_Toc78984980)

[6.4.3 Error Results 14](#_Toc78984981)

[6.5 Get Transfer Information (api/TransferLogs) 14](#_Toc78984982)

[6.5.1 Parameters 14](#_Toc78984983)

[6.5.2 Success Results 14](#_Toc78984984)

[6.5.3 Error Results 16](#_Toc78984985)

[6.6 Get GIS Account Holders (api/TransferLogs) 16](#_Toc78984986)

[6.6.1 Parameters 16](#_Toc78984987)

[6.6.2 Response Codes 16](#_Toc78984988)

[7. Post APIs 16](#_Toc78984989)

[7.1 General POST API Behavior 16](#_Toc78984990)

[7.1.1 Response Codes 17](#_Toc78984991)

[7.1.2 Error Results 17](#_Toc78984992)

[7.2 Post Transfer Requests (api/TransferRequests) 17](#_Toc78984993)

[7.2.1 Parameters 17](#_Toc78984994)

[7.2.2 Post Contents 17](#_Toc78984995)

[7.2.3 Success Results 18](#_Toc78984996)

[7.2.4 Error Results 18](#_Toc78984997)

[7.3 Post Reserve Transfer Requests (api/ReserveRequests) 19](#_Toc78984998)

[7.3.1 Parameters 19](#_Toc78984999)

[7.3.2 Post Contents 19](#_Toc78985000)

[7.3.3 Success Results 19](#_Toc78985001)

[7.3.4 Error Results 20](#_Toc78985002)

[7.4 Post Retail Transfer Requests (api/RetailRequests) 20](#_Toc78985003)

[7.4.1 Parameters 20](#_Toc78985004)

[7.4.2 Post Contents 20](#_Toc78985005)

[7.4.3 Success Results 21](#_Toc78985006)

[7.4.4 Error Results 21](#_Toc78985007)

[7.5 Post Transfer Actions (api/TransferActions) 21](#_Toc78985008)

[7.5.1 Parameters 22](#_Toc78985009)

[7.5.2 Post Contents 22](#_Toc78985010)

[7.5.3 Success Results 22](#_Toc78985011)

[7.5.4 Error Results 22](#_Toc78985012)

[8. Appendix A: NEPOOL GIS Programs 23](#_Toc78985013)

[9. Appendix B: Fuel Type Codes 24](#_Toc78985014)

[10. Appendix C: Project Status Names 25](#_Toc78985015)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | Description | Updated By |
| 1.0 | 8/03/2016 | * Initial Version | Adam Barrett |
| 1.1 | 08/09/2016 | * Made updates to the Authentication Protocol Section | John Shewchuk |
| 1.2 | 11/07/2016 | * Add “Previous Owner” to Get Transferable Positions request (7.2, 7.2.2) * Add “Certificate Serial Number”, “Jurisdiction” and “Date Initiated” to Get Pending Transfers request (7.3, 7.3.2) * Add “Jurisdiction” to Get Inter-Account Transfer Report request (7.5, 7.5.2) * Add “City” and “Status” to Get Generator Information request (7.4, 7.4.2) * Add all generators associated with an account to the Get Generator Information request (7.4) | Devon Walton |
| 1.3 | 12/07/2016 | * Added API status codes for Get Generator information to Appendix C | Adam Barrett |
| 1.4 | 5/10/2017 | * Add VT Tier I and VT Tier II Programs * Add Marine Thermal fuel type | James Webb |
| 2.0 | 12/17/2018 | * Adding Reserve and Retail SubAccount transfer types * Aligning terminology and field names with Swagger definitions * Fixing representation of response codes / structure * Adding support for a single login to be used to access the API of multiple GIS accounts * Added new CT Class I FERC and MA CES state eligibilities to Appendix A | Andrew Thornton |
| 2.1 | 12/28/2018 | * Added support for new optional fields to be included on certificate transfers:   + *notes*   + *pricePerCertificate* * Added the new NH Class I Biodiesel Producer state eligibility to Appendix A | Joel O’Neil |
| 2.2 | 4/15/2021 | * Formatting Updates | Joe Varnas Jr |
| 2.3 | 07/22/2021 | * Added Get Account Holders | Joe Varnas Jr |
| 2.4 | 08/04/2021 | * Updated Section 4.1.1 Header (Production and UAT) * Updated Appendix A: NEPOOL GIS Programs * Updated Appendix B: Fuel Type Codes | Mary Frantz |

# Summary

This document describes the programmatic interfaces (APIs) for the NEPOOL-GIS Registry from a business user perspective. This document will explain how to create an API Login for your account, the rules governing the use of the API, the performance expectations, and validations that occur on incoming data and request parameters.

This document also contains basic instructions on how to use Swagger to get the technical details for each API, including all input parameters and output data set structures.

# Creating an API Login

NEPOOL-GIS Accounts will have dedicated logins for accessing the APIs. Regular user logins will not be able to authenticate through the API.

API Logins can be created by the designated Account Manager through the NEPOOL-GIS Interface.

## API Login Creation Interface

1. Account Manager logs in to GIS website.
2. In the Account Management module on the left-hand side, click the link for “API Management.” This will take the user to the APX Management interface.
3. Enter a Login Name and Password, and Confirm Password, then click the “Save” button. At this point the API Login will be saved, but not activated. The GIS Administrator will be notified and will proceed with the activation process.
   * Passwords must be at least 6 characters long, contain at least one alphabetic character, one numeric character, and one special character. The following special characters are allowed: @ + - $ \* # % ~ = \_ !
   * Login name conforms to formatting rules and is available (not already in use).
4. Account Holders must submit IP Addresses and Subnet Masks for the machines which will be accessing the API. Account Holder may enter up to 10 IP Addresses, but these must be approved by the GIS Administrator before the machines may access the API. The GIS Administrator is notified when IP Addresses are added.
   * IP Address and Subnet Mask support IPv4 and IPv6. Inputs must be correctly formatted for one of those versions.

# General API Information

## Authentication

The API user will authenticate using the API Login Name and Password created in the API Management screen that is part of the Account Management module on the dashboard of the registry web application.

Client API consumers will authenticate against an OAuth2 endpoint exposed by the client API (see endpoint URLs below). The OAuth2 endpoint acts as the authorization server for your client and will provide the granted credentials for access to the API Endpoints. This Authorization API POST request will return a short-lived JSON Web Token that will be provided in calls to the application endpoints exposed by the API.

**Production Authentication Endpoint:** https://apxjwtauthprod.apx.com/oauth/token

**UAT Authentication Endpoint:** <https://apxjwtauthuat.apx.com/oauth/token>

### Headers

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Description** |
| UAT Authorization | Basic  TkVQT09MLUNsaWVudC1BUEk6OVJAV3lEPnhIUjlfSU5tc1dWa2s= | This identifies you as a NEPOOL GIS client to the authorization UAT server |
| Production Authorization | Basic TkVQT09MLUNsaWVudC1BUEk6cFleQ0FxSHVQeiZiN3olLXNISnAyPVF6M3EhV0FXWC10QnUmUGNwQQ== | This identifies you as a NEPOOL GIS client to the authorization Production server |

### Parameters

|  |  |
| --- | --- |
| **Field** | **Description** |
| Username | Client API Service User Name |
| Password | Client API Service Password |
| grant\_type | Value: password  [This is associated with the OAuth2 password credentials scenario](https://tools.ietf.org/html/rfc6749#section-4.3.2) |

### Results

|  |  |
| --- | --- |
| **Field** | **Description** |
| access\_token | This is the token to be used in the “bearer” value of the HTTP Authorization header |
| token\_type | The type of the token to be used in the API Requests.  This value returned will be “Bearer” |
| expires\_in | Duration in which the token will expire and a subsequent authentication request will need to be made if time expires. |

Attempts to call application endpoints without a valid token will result in an HTTP error message being returned.

### Status Codes

|  |  |  |
| --- | --- | --- |
| **HTTP Status Code** | **Status** | **Status Message** |
| 200 | SUCCESS | Successfully authenticated |
| 401 | ERROR | Bad Request  Invalid Login ID or Password |

## Get / Post Security

Present the authorization header below to call into the APIs.

### Headers

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Description** |
| Authorization | Bearer {access\_token} | The access\_token that is returned from the authentication request will be inserted into the value field. |

## Acceptable Use

The initial Acceptable Use Policy will be four calls per endpoint per IP Address per minute. Endpoints are Get Positions, Get Transfers (pending), Get Generators, Get Transfer Logs, Post Transfers, and Post Transfer Actions. Violation of the acceptable use policy will result in an error being returned to API caller (HTTP 429) indicating that the acceptable use policy has been violated.

## Auxiliary/Reference Data

Some of the data sent and received through the API uses codes to represent certain field values. These codes and what they represent must be communicated to API users.

* **Account IDs** – Available through the “GIS Account Holders” public report
* **SubAccount IDs** – Available in the “Subaccounts Summary” section of the Account Status module
* **NEPOOL GIS Programs** – [Appendix A](#_Appendix_A:_NEPOOL_1)
* **Fuel Type Codes** – [Appendix B](#_Appendix_B:_Error)
* **NEPOOL Project Status** – [Appendix C](#_Appendix_D:_Status)

## Automated Emails

Accounts with an active API login will not receive email notifications for transfers/actions that they initiate.

The current email behavior for actions initiated by a counterparty will not change. Currently, counterparty actions (transfers/confirms/rejects/withdraws), generate one email per action. When a user takes any single action that involves multiple blocks of instruments, the Registry sends a single email associated with the action, and not individual emails for the constituent instrument blocks. This will also be the case for accounts that use the API.

## Event Logging

Event Logging for user actions performed through the API should be the same as performing those actions through the User Interface.

Specifically, initiating Transfers and acting on Transfers (Confirming, Rejecting, Withdrawing) will be logged and will be accessible to the User in My Event Log.

## User Acceptance Testing (UAT) and Production Base URLs

To call a specific API, append “api/*method name*” to the following base URLs

User Acceptance Testing (UAT) -> <https://gis-app-uat01.apx.com/clientapi2/>

Production -> <https://www1.nepoolgis.com/clientapi2/>

# Swagger Technical API Specification

NEPOOL GIS uses the Swagger specification to describe the integration API endpoints available to consumers. A swagger.json file is provided (available on registry website; see links below) that contains the definitions of the endpoints that will allow you to become familiar with the requests and responses provided. You can follow the steps outlined below to get started. There is also a Swagger UI page available through the registry website (see links below)

**Swagger.json file:** [**Production**](https://www1.nepoolgis.apx.com/clientapi2/v2/api-docs?group=nepool-client-api) **|** [**UAT**](https://gis-app-uat01.apx.com/clientapi2/v2/api-docs?group=nepool-client-api)

**Swagger UI Page:** [**Production**](https://www1.nepoolgis.apx.com/clientapi2/swagger-ui.html) **|** [**UAT**](https://gis-app-uat01.apx.com/clientapi2/swagger-ui.html)

## View GIS APIs on Swagger

1. Go to [Swagger website](http://swagger.io/). This site is the community site that describes the swagger specification and has demonstrations and downloads available
2. Go to the [“demo” area](http://editor.swagger.io/#/). This will take you to a hosted solution where you can view the definitions and generate servers and clients in many mainstream languages. You will need to create the client code for consumption of the endpoints. The server generated code can be used to create stubs to simulate interactions with the live endpoint.
3. Upload the GIS\_Swagger\_API.json file (see links, above) to the swagger editor
   1. Go to “File” menu item
   2. Go to the “Import File” menu item
   3. Navigate to the file location
   4. Upload the file
4. You can now view the GIS API definitions in the right-hand pane. (The “Warnings” can be ignored as they are alerting you to a non-standard description field that is generated)

## Generate Client for GIS APIs

1. Perform the “View GIS APIs” as described above.
2. Select the “Generate Client” menu item.
3. Select your language of choice and download the SDK
   1. This will download an SDK in your language of choice. Please note that you may need to make modifications to the toolkit (e.g. Username / Passwords, Endpoint URL changes, et al).
   2. This can serve as a starting point to setting up your code to consume the APIs.

## Generate Server for GIS APIs

1. Perform the “View GIS APIs” as described above.
2. Select the “Generate Server” menu item.
3. Select your language of choice.
   1. This will download an SDK in your language of choice. Please note that you may need to make modifications to the toolkit (e.g. Username / Passwords, Endpoint URL changes, et al).
   2. This can serve as a starting point to setting up the server API stubs.

# Get APIs

## General GET API Behavior

* If no data is found for request, empty dataset is returned AND a Success code (HTTP 200).
* Result sets/Responses limited to 60,000 lines. If over 60k, an error will be returned (HTTP 400).

### Response Codes

|  |  |  |
| --- | --- | --- |
| **HTTP Status Code** | **Status** | **Status Message** |
| 200 | SUCCESS |  |
| 400 | ERROR | Invalid parameter(s) |
| 401 | ERROR | Unauthorized access |
| 500 | ERROR | An unexpected error has occurred |

### Error Results

If the request fails validation, then the following structure will be returned. This will correspond to an HTTP Status Code = 400, 401, or 500. That return set includes the following:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| errors | Container for errors | N/A |
| errors {corellationId} | NULL for Get methods | String |
| errors {message} | String corresponding to the specific error encountered for the corresponding Row ID. *See the individual Get methods for context specific errors.* | String |
| errors {parameterName} | The field name of the parameter that failed | String |
| operationId | GUID associated with system logging. Provide when requiring APX support. | String |

## Get Transferable Positions (api/Position)

A request to retrieve the list of certificate blocks that are eligible for transfer. This includes all of the certificates belonging to the API user that are Transferable, Banked, or on the Bulletin Board.

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| year | [YYYY] | Int | No (Required if Vintage Month is provided) |
| month | [MM] | Int | No (Required if Vintage Year is provided) |
| ahIds\*\* | GIS Account ID | Int | No |

\*If no Vintage parameters are provided, all vintages will be returned.

\*\*Multiple Account IDs can be provided. If no Account ID(s) are provided then information for all accounts will be returned.

### Success Results

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| ahId | GIS Account ID | Int |
| unitId | GIS Unit ID | String |
| generatorName | Concatenation of “Plant Name” + “ - “ + “Unit Name”, e.g. “Plant1 – Unit4” | String |
| jurisdiction | State or Province, e.g. "MA", "CT" | String |
| fuelType | Code for fuel type (See [Appendix B](#_Appendix_B:_Error)) | String |
| vintage {year} | YYYY | Int |
| vintage {month} | MM | Int |
| certificateSerialNumberRange | Serial Number sequence created by GIS, e.g. "599730 - 1 to 100" | String |
| quantity | Quantity of Certificates | Int |
| eligibilities | Delimited list of programs for which the certificate(s) are eligible (See [Appendix A](#_Appendix_A:_NEPOOL_1)) | String |
| previousOwner | Account ID of previous Certificate owner | Int |

### Error Results

If the request fails validation, then the following structure will be returned. This will correspond to an HTTP Status Code = 400. That return set includes the following:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| errors {Message} | String corresponding to the specific error encountered for the corresponding Row ID. Possible validation errors include:   * Results exceed 60,000 records * Vintage Year/Vintage Month improperly formatted * Vintage Month not provided with Vintage Year * Vintage Year not provided with Vintage Month | String |

## Get Pending Transfers (api/Transfers)

A request to retrieve all pending certificate transfers, both incoming and/or outgoing. These are the transfers initiated by a counterparty (incoming) or account holder (outgoing) that are waiting on the account holder to confirm, reject, or withdraw. *This method only applies to Another Account Holder transfers since other types of transfers are synchronously completed and don’t require an “action”.*

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| ahIds\* | GIS Account IDs | INT | No |
| direction\*\* | “I” or “O” for incoming or outgoing | String (1) | No |

\*Multiple Account IDs can be provided. If no Account ID(s) are provided then information for all accounts will be returned.

\*\*If no Direction input parameter is provided, both Incoming and Outgoing Transfers will be returned.

### Success Results

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| ahId | GIS Account ID | Int |
| certificateSerial NumberRanges | Serial Number sequence of pending Certificates, e.g. "599730 - 1 to 100" | String |
| dateInitiated | Date and time the pending transfer was initiated, e.g. “6/5/2008 17:53” | Datetime |
| direction | “I” for incoming pending transactions or “O” for outgoing pending transactions. | String |
| eligibilities | Delimited list of programs for which the certificate(s) are eligible (See [Appendix A](#_Appendix_A:_NEPOOL_1)) | String |
| fuelType | Code for fuel type (See [Appendix B](#_Appendix_B:_Error)) | String |
| generatorName | Concatenation of “Plant Name” + “ - “ + “Unit Name”, e.g. “Plant1 – Unit4” | String |
| jurisdiction | State or Province, e.g. "MA", "CT" | String |
| notes | Optional text field that allows the user to enter additional transfer details | String |
| pricePerCertificate | Optional field for the user to enter the price associated with the certificates in a single transaction | Number |
| quantity | Quantity of Certificates | Int |
| transfereeId | Transferee Account ID | Int |
| transferId | System-generated ID for the transfer | String |
| transferorId | Transferor Account ID | Int |
| unitId | GIS Unit ID | String |
| vintage {month} | MM | Int |
| vintage {year} | YYYY | Int |

### Error Results

If the request fails validation, then the following structure will be returned. This will correspond to an HTTP Status Code = 400. That return set includes the following:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| errors {message} | String corresponding to the specific error encountered for the corresponding Row ID. Possible validation errors include:   * Results exceed 60,000 records * Invalid direction input | String |

## Get Generator Information (api/Generators)

A request to retrieve data for all generators associated with the account, regardless of a project’s status.

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| ahIds\* | GIS Account IDs | Int | No |
| unitIds\*\* | GIS Unit IDs | String | No |

\*Multiple Account IDs can be provided. If no Account ID(s) are provided then information for all accounts will be returned.

\*\*Multiple Unit IDs may be provided. If no input parameters are provided, information for all generators will be returned.

### Success Results

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| ahId | GIS Account ID | Int |
| city | City | String |
| eligibilities | Delimited list of programs for which the generator is eligible (See [Appendix A](#_Appendix_A:_NEPOOL_1)) | String |
| fuelType | Code for fuel type (See [Appendix B](#_Appendix_B:_Error)) | String |
| generatorName | Concatenation of “Plant Name” + “ - “ + “Unit Name”, e.g. “Plant1 – Unit4” | String |
| gisRegistrationDate | MM/DD/YYYY HH:mm | Datetime |
| jurisdiction | State or Province, e.g. "MA", "CT" | String |
| nameplateCapacity | Unit’s name plate capacity | Decimal |
| status | Current status of the Generator (See [Appendix C](#_Appendix_D:_Status)) | String |
| unitId | GIS Unit ID | String |
| vintage {month} | MM | Int |
| vintage {year} | YYYY | Int |

### Error Results

If the request fails validation, then the following structure will be returned. This will correspond to an HTTP Status Code = 400. That return set includes the following:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| errors {message} | String corresponding to the specific error encountered for the corresponding Row ID. Possible validation errors include:   * Results exceed 60,000 records * Invalid Unit ID passed | String |

## Get Transfer Information (api/TransferLogs)

A request to retrieve the Transfer information.

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| ahIds\* | GIS Account IDs | Int | No |
| fromDate | From Activity Date [YYYY-MM-DD] | Date | Yes |
| toDate | To Activity Date [YYYY-MM-DD] | Date | Yes |
| transferType\*\* | anotherAccountHolder, reserve, retailSubAccount | String (32) | No |
| action\*\* | Transfer, Confirm, Reject, Withdraw | String (10) | No |

\*Multiple Account IDs can be provided. If no Account ID(s) are provided then information for all accounts will be returned.

\*\*Only one Transfer Type parameter may be provided. When no Transfer Type is provided, all transfer types within the activity dates will be returned.

\*\*\*Only one Action parameter may be provided. When no Action parameter is provided, all actions within the activity dates will be returned. All Reserve and Retail SubAccount will have an action of “Confirm”.

### Success Results

Returns data for transfers that occurred within the date range specified in the request. Transfer data includes information on transfer initiation, as well as when transfers are Confirmed/Rejected/Withdrawn. As such, Confirmed/Rejected/Withdrawn transfers will have multiple rows of data.

The following information will be provided in the request response:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| action | Type of Transfer Action:   * confirm * reject * transfer * withdraw | String |
| actionTime | Date and time that transfer was executed, e.g. “6/5/2008 17:53” | Datetime |
| ahId | GIS Account ID | Int |
| certificateSerial NumberRanges | Serial Numbers of transferred certificates, e.g. "599730 - 1 to 100". | String |
| eligibilities | Delimited list of programs for which the generator is eligible (See [Appendix A](#_Appendix_A:_NEPOOL_1)) | String |
| fuelType | Code for fuel type (See [Appendix B](#_Appendix_B:_Fuel)) | String |
| generatorName | Concatenation of “Plant Name” + “ - “ + “Unit Name”, e.g. “HOOSAC - HOOSAC WIND” (“HOOSAC”=Plant Name, “HOOSAC WIND”=Unit Name) | String |
| jurisdiction | State or Province, e.g. "MA", "CT" | String |
| loginName | Login name of user that took the action, e.g. “ptoomey”. May be an Account Holder user, or a counterparty user | String |
| notes | Optional text field that allows the user to enter additional transfer details | String |
| pricePerCertificate | Optional field for the user to enter the price associated with the certificates in a single transaction | Number |
| quantity | Quantity of Certificates | Int |
| reason | The reason, or on whose behalf, the Reserve Transfer was performed  N/A for Another Account Holder and Retail SubAccount transfers | String |
| reserveVoluntary | Whether or not the Reserve Transfer was for voluntary purposes  N/A for Another Account Holder and Retail SubAccount transfers | Bit |
| retailSubAccountId | SubAccount destination for Retail SubAccount transfers.  N/A for Another Account Holder and Reserve transfers | Int |
| retirementState | The NEPOOL Member state in which the Reserve or Retail SubAccount transfer is effective  N/A for Another Account Holder transfers | String |
| transfereeId | Account ID of Buying Account  N/A for Reserve and Retail SubAccount transfers | String |
| transferID | System-generated ID for the transfer  N/A for Reserve and Retail SubAccount transfers | String |
| transferorId | Account ID of Selling Account  N/A for Reserve and Retail SubAccount transfers | String |
| transferType | Type of Transfer:   * anotherAccountHolder, reserve, retailSubAccount | String |
| unitID | GIS Unit ID | String |
| vintage {month} | MM | Int |
| vintage {year} | YYYY | Int |

### Error Results

If the request fails validation, then the following structure will be returned. This will correspond to an HTTP Status Code = 400. That return set includes the following:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| errors {message} | String corresponding to the specific error encountered for the corresponding Row ID. Possible validation errors include:   * Results exceed 60,000 records * Invalid status passed * Both FromDate and ToDate must be passed * ToDate must be greater than or equal to FromDate | String |

## Get GIS Account Holders (api/AccountHolder)

* A request to retrieve the GIS Account Holders (Public Report)

### Parameters

None

### Success Results

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| ahId | GIS Account ID | Int | Yes |
| city | GIS Account Holder City | String | No |
| fax | GIS Account Holder Fax Number | String | No |
| name | GIS Account Holder Name | String | No |
| phone | GIS Account Holder Phone Number | String | No |
| representative | GIS Account Holder Representative | String | No |
| state | GIS Account Holder State | String | No |
| streetAddress1 | GIS Account Holder Street Address 1 | String | No |
| streesAddress2 | GIS Account Holder Street Address 2 | String | No |
| Website | GIS Account Holder Website | String | No |
| zipCode | GIS Account Holder Zip Code | String | No |

### Response Codes

|  |  |  |
| --- | --- | --- |
| **HTTP Status Code** | **Status** | **Status Message** |
| 200 | SUCCESS |  |
| 400 | ERROR | Invalid parameter(s) |
| 401 | ERROR | Unauthorized access |
| 500 | ERROR | An unexpected error has occurred |

# Post APIs

## General POST API Behavior

* If any data within a post data set fails validation, the entire file will be rejected.
* Posts limited to 20,000 lines; post over that threshold will be rejected.

### Response Codes

|  |  |  |
| --- | --- | --- |
| **HTTP Status Code** | **Status** | **Status Message** |
| 200 | SUCCESS |  |
| 400 | ERROR | Invalid parameter(s) |
| 401 | ERROR | Unauthorized access |
| 413 | ERROR | Request entity too large |
| 500 | ERROR | An unexpected error has occurred |

### Error Results

If the request fails request validation, then the following structure will be returned. This will correspond to an HTTP Status Code = 400, 401, 413, or 500. That return set includes the following:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| errors | Container for errors | N/A |
| errors {corellationId} | The row number corresponding to the error message | String |
| errors {message} | String corresponding to the specific error encountered for the corresponding correlation ID. *See the individual Post methods for context specific errors.* | String |
| errors {parameterName} | The field name of the parameter that failed | String |
| operationId | GUID associated with system logging. Provide when requiring APX support. | String |

## Post Transfer Requests (api/TransferRequests)

This API is for initiating Another Account Holder Transfers of certificates that are Transferable, Banked, or on the Bulletin Board.

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| ahId\* | GIS Account ID | Int | No |

\*Only a single Account ID can be provided. If no Account ID is provided then the primary account (i.e. GIS account that created the API credentials) is assumed.

### Post Contents

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| buyerAccountId | Buyer Account ID | Int | No |
| certificateSerialNumberRange | Serial Number sequence created by GIS, e.g. "599730 - 1 to 100". User may not consolidate adjacent certificate serial numbers—If there are two Certificate blocks with consecutive serial numbers, they must be submitted as separate rows. | String | Yes |
| notes | Optional text field that allows the user to enter additional transfer details | String | No |
| pricePerCertificate | Optional field for the user to enter the price associated with the certificates in a single transaction | Number | No |
| quantity | Quantity of Certificates. Must be within the range of Serial Numbers | Int | Yes |
| requestCorrelationId | External ID provided by the user, to be used when returning the method results to identifying specific Transfers (“collections”). Will not persist with data. | String | Yes |

Note: The Certificate Serial Number Range is not something that can be manipulated by the API user. Only serial number ranges provided by the API should be used by the API user when communicating transfer instructions.

### Success Results

If all rows pass validation, then the transfers are completed and a result set will be returned that includes the following:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| requestCorrelationId | External ID provided by the user when submitting | String |
| resultCodes | NULL | String |
| transferId | System-generated ID for the transfer | String |
| remainingCertificateSerialNumberRange | Certificate Serial Numbers for the holding included in the transfer that remain in the user’s account after the transfer. If the same serial numbers are submitted for multiple transfers in a single batch, this field should reflect the remaining serial numbers after all transfers in the batch are completed. This field will be empty if no certificates for the serial numbers used in the transfer remain. | String |

### Error Results

If one of the rows in the request fails business validation then no transfers are initiated within the system, and a dataset is returned with all rows that failed validation. This will correspond to an HTTP Status Code = 400. That return set may include the following error messages:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| errors {message} | String corresponding to the specific error encountered for the corresponding Row ID. Possible validation errors include:   * Missing/invalid Row Identifier * Missing/incorrect Certificate Serial Numbers * Missing/invalid quantity * Quantity greater than available certificates * Sum of quantity greater than available certificates * Missing/incorrect Buyer Account ID * Certificates invalid for transfer | String |

## Post Reserve Transfer Requests (api/ReserveRequests)

This API is for initiating Reserve Transfers of certificates that are Transferable, Banked, or on the Bulletin Board.

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| ahId\* | GIS Account ID | Int | No |

\*Only a single Account ID can be provided. If no Account ID is provided then the primary account (i.e. GIS account that created the API credentials) is assumed.

### Post Contents

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| requestCorrelationId | External ID provided by the user, to be used when returning the method results to identifying specific Transfers (“collections”). Will not persist with data. | String | Yes |
| certificateSerial NumberRange | Serial Number sequence created by GIS, e.g. "599730 - 1 to 100". User may not consolidate adjacent certificate serial numbers—If there are two Certificate blocks with consecutive serial numbers, they must be submitted as separate rows. | String | Yes |
| quantity | Quantity of Certificates. Must be within the range of Serial Numbers | Int | Yes |
| reason | The reason, or on whose behalf, associated with the Reserve Transfer | String | Yes |
| reserveVoluntary | Whether or not the Reserve Transfer was for voluntary purposes | Bit | Yes |
| retirementState | The NEPOOL Member state in which the retirement is effective | String | No |

Note: The Certificate Serial Number Range is not something that can be manipulated by the API user. Only serial number ranges provided by the API should be used by the API user when communicating transfer instructions.

### Success Results

If all rows pass validation, then the transfers are completed and a result set will be returned that includes the following:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| requestCorrelationId | External ID provided by the user when submitting | String |
| resultCodes | NULL | String |
| transferId | System-generated ID for the transfer | String |
| remainingCertificateSerialNumberRange | Certificate Serial Numbers for the holding included in the transfer that remain in the user’s account after the transfer. If the same serial numbers are submitted for multiple transfers in a single batch, this field should reflect the remaining serial numbers after all transfers in the batch are completed. This field will be empty if no certificates for the serial numbers used in the transfer remain. | String |

### Error Results

If one of the rows in the request fails business validation then no transfers are initiated within the system, and a dataset is returned with all rows that failed validation. This will correspond to an HTTP Status Code = 400. That return set may include the following error messages:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| errors {message} | String corresponding to the specific error encountered for the corresponding Row ID. Possible validation errors include:   * Missing/invalid Row Identifier * Missing/incorrect Certificate Serial Numbers * Missing/invalid quantity * Quantity greater than available certificates * Sum of quantity greater than available certificates * Missing/invalid Reason attribute * Missing/invalid Voluntary attribute * Missing/invalid State attribute * Certificates invalid for transfer | String |

## Post Retail Transfer Requests (api/RetailRequests)

This API is for initiating Retail SubAccount Transfers of certificates that are Transferable, Banked, or on the Bulletin Board.

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| AhId\* | GIS Account ID | Int | No |

\*Only a single Account ID can be provided. If no Account ID is provided then the primary account (i.e. GIS account that created the API credentials) is assumed.

### Post Contents

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| requestCorrelationId | External ID provided by the user, to be used when returning the method results to identifying specific Transfers (“collections”). Will not persist with data. | String | Yes |
| certificateSerial NumberRange | Serial Number sequence created by GIS, e.g. "599730 - 1 to 100". User may not consolidate adjacent certificate serial numbers—If there are two Certificate blocks with consecutive serial numbers, they must be submitted as separate rows. | String | Yes |
| quantity | Quantity of Certificates. Must be within the range of Serial Numbers | Int | Yes |
| retailSubAccountId | SubAccount destination for Retail SubAccount transfers. | Int | Yes |

Note: The Certificate Serial Number Range is not something that can be manipulated by the API user. Only serial number ranges provided by the API should be used by the API user when communicating transfer instructions.

### Success Results

If all rows pass validation, then the transfers are completed and a result set will be returned that includes the following:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| requestCorrelationId | External ID provided by the user when submitting | String |
| resultCodes | NULL | String |
| transferId | System-generated ID for the transfer | String |
| remainingCertificateSerialNumberRange | Certificate Serial Numbers for the holding included in the transfer that remain in the user’s account after the transfer. If the same serial numbers are submitted for multiple transfers in a single batch, this field should reflect the remaining serial numbers after all transfers in the batch are completed. This field will be empty if no certificates for the serial numbers used in the transfer remain. | String |

### Error Results

If one of the rows in the request fails business validation then no transfers are initiated within the system, and a dataset is returned with all rows that failed validation. This will correspond to an HTTP Status Code = 400. That return set may include the following error messages:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| errors {message} | String corresponding to the specific error encountered for the corresponding Row ID. Possible validation errors include:   * Missing/invalid Row Identifier * Missing/incorrect Certificate Serial Numbers * Missing/invalid quantity * Quantity greater than available certificates * Sum of quantity greater than available certificates * Missing/invalid Retail SubAccount ID * Certificates invalid for transfer | String |

## Post Transfer Actions (api/TransferActions)

This API is for acting on Certificates in a Pending state (either incoming or outgoing). Note that actions for incoming and outgoing transfers can be included in the same submission. *This method only applies to Another Account Holder transfers since other types of transfers are synchronously completed and don’t require an “action”.*

### Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| ahId\* | GIS Account ID | Int | No |

\*Only a single Account ID can be provided. If no Account ID is provided then the primary account (i.e. GIS account that created the API credentials) is assumed.

### Post Contents

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Description** | **Data Type** | **Required?** |
| transferId | System-generated ID for the transfer | String | Yes |
| action | Action to take on the Transaction:   * Withdraw (Outgoing transfers only) * Confirm (Incoming transfers only) * Reject (Incoming transfers only) | String | Yes |

### Success Results

If all rows pass validation, then the transfer actions are completed and a result set will be returned that includes the following:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| transferId | System-generated ID for the transfer | String |
| resultCodes | NULL | String |

### Error Results

If one of the rows in the request fails business validation then no transfer actions are initiated within the system, and a dataset is returned with all rows that failed validation. This will correspond to an HTTP Status Code = 400. That return set may include the following error messages:

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | **Data Type** |
| errors {message} | String corresponding to the specific error encountered for the corresponding Row ID. Possible validation errors include:   * Missing/invalid Transfer ID * Invalid Action * Invalid Action for Transfer ID * Duplicate Transfer ID passed | String |

# Appendix A: NEPOOL GIS Programs

The following programs are currently returned as columns when running reports for Transferable and Pending Certificate Transfers. These may be represented in the API by programs codes.

|  |  |
| --- | --- |
| Program Name | Program Code |
| CT CEO | CTCEO |
| CT Class I | CTClassI |
| CT Class I FERC | CTClassIFERC |
| CT Class II | CTClassII |
| CT Class III | CTClassIII |
| CT LREC | CTLREC |
| Eligible MA NOx Allowances | MARenewableNOx |
| Green-E | GreenE |
| Low Impact Hydro Institute | LIHI |
| MA APS Alternative Generation Unit | MAApsAlternative |
| MA Auction Solar Carve-Out I Unit | MAAuctionSolarCarveOutI |
| MA Auction Solar Carve-Out II Unit | MAAuctionSolarCarveOutII |
| MA CES | MACES |
| MA CES E | MACES\_E |
| MA RPS Class I Renewable Generation Unit | MANewRenewable |
| MA RPS Class II Renewable Generation Unit | MARPSClassIIRenewable |
| MA RPS Class II Waste Energy Generation Unit | MARPSClassIIWasteEnergy |
| MA Solar Carve-Out I Unit | MASolarCarveOutI |
| MA Solar Carve-Out II Unit | MASolarCarveOut2 |
| MA Solar Carve-Out II Unit | MASolarCarveOutII |
| ME Class I | MEClassI |
| ME Class II | MEClassII |
| ME Class IA | MEClassIA |
| ME CO2 Netting | MECO2Netting |
| ME Community Based Renewable Energy | MECommunityBasedRenewableEnergy |
| ME Ren/Eff Energy Source | MERenEffSource |
| ME TREC | METRec |
| NH Class I | NHClassI |
| NH Class I Thermal | NHClassIThermal |
| NH Class II | NHClassII |
| NH Class I Biodiesel Producer | NHClassIBiodieselProducer |
| NH Class III | NHClassIII |
| NH Class IV | NHClassIV |
| RI Existing Renewable Resource | RIExistingRenewable |
| RI New Renewable Resource | RINewRenewable |
| VT Tier I | VTTierI |
| VT Tier II | VTTierII |

# Appendix B: Fuel Type Codes

|  |  |  |
| --- | --- | --- |
| Fuel Type Code | Short Description | Description |
| ASH | Air-source heat pump | Air-source heat pump |
| BIG | Biogas | Biogas |
| BIM | Biomass | Biomass |
| BO1 | Biodiesel1 | 100% neat |
| BO2 | Biodiesel2 | Less than 100% neat |
| CLM | CLM | Conservation and Load Management |
| CO1 | Coal | Coal |
| CP1 | Composite | Composite |
| DG1 | Digester gas | Digester gas |
| DGH | Deep geothermal heat exchange | Deep geothermal heat exchange |
| DI1 | Diesel | Diesel |
| DRP | DRP | Curtailment-based Demand Response |
| EN1 | Energy Efficient Steam | Energy Efficient Steam |
| EN2 | Energy Storage | Energy Storage |
| ER1 | Efficient Resource (Maine) | Efficient Resource (Maine) |
| ET1 | Ethanol | Ethanol |
| FLC | Fuel cell | Fuel cell |
| GA1 | Gasification | Gasification |
| GE1 | Geothermal | Geothermal |
| GWH | Ground- and Water-source heat pump | Ground- and Water-source heat pump |
| H2O | Hydroelectric/Hydropower | Hydroelectric/Hydropower |
| HY1 | Hydrokinetic | Hydrokinetic |
| JET | Jet | Jet |
| LBL | Liquid biofuels | Liquid biofuels |
| LEC | Low Emission (Connecticut) | Low emission advanced renewable energy conversion technologies |
| LG1 | Landfill gas | Landfill gas |
| MSW | Municipal solid waste | Municipal solid waste |
| MT1 | Methanol | Methanol |
| MTH | Marine Thermal | Marine Thermal |
| NG1 | Natural Gas | Natural Gas |
| NU1 | Nuclear | Nuclear |
| OC1 | Ocean Thermal | Thermal |
| OC2 | Ocean Wave | Wave |
| OC3 | Ocean Tidal | Tidal |
| OC4 | Ocean1 | Movement or the latent heat of the ocean |
| OC5 | Ocean2 | Ocean Current |
| OIL | Oil | Oil |
| PA1 | Paper-derived | Paper-derived |
| PS1 | Pumped Storage | Pumped Storage |
| SO1 | Solar Thermal | Thermal |
| SO2 | Solar Photovoltaic | Photovoltaic |
| TE1 | Trash-to-energy | Trash-to-energy |
| WA1 | Waste Energy | Waste Energy |
| WND | Wind | Wind |
| WO1 | Waste Oil | Waste Oil |
| WOD | Wood | Wood |

# Appendix C: Project Status Names

|  |  |
| --- | --- |
| API Status Code | Status Name |
| ACT | Approved |
| INA | Inactive |
| NIN | Need Info |
| PEN | Pending |
| REJ | Rejected |