



Catalist API

Developer Guide



Contents

1	Getting started	4
1.1	What is Catalist?	4
1.2	What is the Catalist API?	4
1.3	Purpose of this document	6
1.4	Document audience and pre-requisites	6
1.5	Getting a Catalist API account	7
1.5.1	Neartime domain names	7
1.5.2	Talking with your Experian Account Manager	7
1.5.3	Setting up your account	7
2	Logging in	9
2.1	About the log in request	9
2.1.1	HTTP summary details	9
2.1.2	Standard header	9
2.1.3	Request body key and value pairs	9
2.1.4	Example log in request	10
2.2	Log in response	10
2.2.1	Successful log in	10
2.2.2	Error response	10
2.2.3	Processing the log in response	10
3	Sending and processing searches	11
3.1	Types of search request	11
3.2	Using a GET search request	11
3.2.1	About the request	11
3.2.2	HTTP summary details	11
3.2.3	Custom headers	12
3.2.4	Query string parameter name and value pairs	12
3.2.5	Example GET search requests	13
3.3	Using a POST search request	13
3.3.1	About the request	13
3.3.2	HTTP summary details	14
3.3.3	Standard header	14
3.3.4	Request body key and value pairs	14
3.3.5	Example POST search requests	15
3.4	Search response (GET and POST)	16
3.4.1	Matched search	16

3.4.2	Non-matched search	18
3.4.3	Error response	18
3.4.4	Processing the response data	19
4	HTTP responses	20
4.1	Introduction	20
4.2	200 (OK) responses	20
4.2.1	200 – Matched search	20
4.2.2	200 – Non-matched search	21
4.2.3	200 – Successful log in	21
4.3	400 – (Bad Request) responses	21
4.3.1	400 – Invalid value for '[name]'.	21
4.3.2	400 – Invalid value for '[name]', must be [valid_values]	22
4.3.3	400 – 'latitude' is out of range, must be between -90 and 90.	22
4.3.4	400 – 'longitude' is out of range, must be between -180 and 180.	23
4.3.5	400 – Missing query parameter '[name]'	23
4.3.6	400 – '[name]' is a required property	24
4.3.7	400 – Request body is not valid JSON	24
4.3.8	400 – Request can not include both postcode and latitude/longitude parameters	25
4.3.9	400 – Request must include either postcode or latitude/longitude parameters	25
4.3.10	400 – '[value]' is not of type '[data_type]'	26
4.4	401 – (Unauthorized) responses	26
4.4.1	401 – Asset Not Found.	26
4.4.2	401 – Invalid token	27
4.4.3	401 – Unauthorized Client	27
4.4.4	401 – Unauthorized Client, null token	28
4.5	404 – (Not Found) response	28
4.6	415 – (Unsupported Media Type) response	29
4.7	429 – (Too Many Requests) response	29
4.8	500 – (Internal Server Error) responses	30
4.8.1	500 – Internal Server Error	30
4.8.2	500 – Problem with authentication	30
4.8.3	500 – Problem with login	30
4.9	500 – (Internal Server Error) response	31
4.10	503 – (Service Unavailable) responses	31
4.10.1	503 – Internal refresh in progress	31
4.10.2	503 – Server error	32
4.11	505 – (HTTP Version Not Supported) response	32

1 Getting started

1.1 What is Catalist?

Experian's Catalist is a leading source of information and analysis on petrol and retail forecourts. With detailed knowledge and understanding of the retail petroleum industry, Catalist helps clients improve their business results today whilst also investing in the future. Catalist's petrol station database holds more than 100 items of site data about every petrol station in the UK and the Isle of Man.

Catalist also provides daily retail fuel prices for all the main fuel grades at sites across the UK. This service enables clients to track forecourt prices, facilitating a clearer understanding of price competition around a site.

Fuel price data is available every day of the year including weekends and Bank Holidays. Fuel prices are updated every day and are made available:

- the following morning. This is known as next-day pricing. For example, the fuel price data for a Monday is available on a Tuesday morning.
- the same day, in the afternoon. This is known as in-day pricing and is a subscription-dependent feature.

The fuel price data available through the API is restricted to the two main grades: unleaded petrol and diesel. Fuel prices for other grades (e.g. premium grades and automotive LPG) are available, but not through the API.

1.2 What is the Catalist API?

The Catalist API is a secure Web-based software interface which allows users to obtain the current Catalist fuel price data (petrol or diesel) for the site within a 2, 5 or 10 mile radius of a specified location. The API is hosted on Experian's Neartime server cluster which offers access to several datasets including Catalist.

The API is primarily intended for clients who want to create their own bespoke mobile or Web-based applications and make use of the Catalist fuel price data. Users of the application specify the required location and a distance from it (2, 5 or 10 miles); the grade of fuel (petrol, diesel, super-diesel or super-unleaded) their vehicle uses; and whether they are interested in the cheapest or the nearest sites to the specified location.

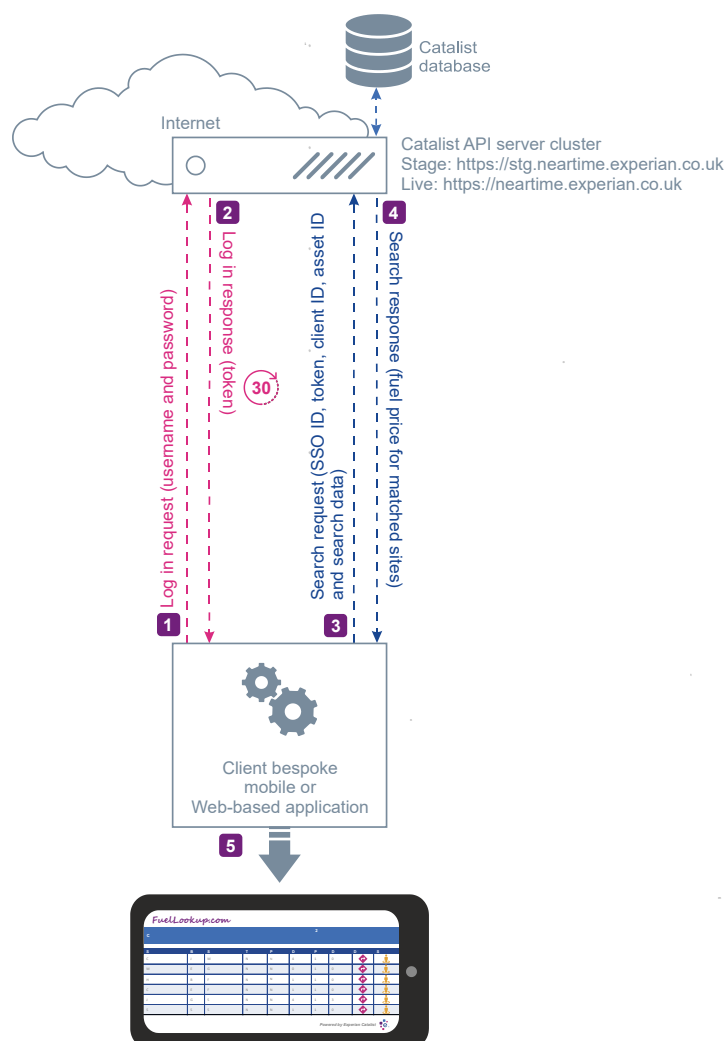
The location can be specified either as a longitude or latitude or a UK postcode. It is expected that the client's application will use location sensing (e.g. via GPS or Wi-Fi triangulation) to determine the user's current location as a longitude and latitude. However, users of the application may also want the option to input a postcode to identify the sites cheapest or nearest to this location. This may be useful when the user only needs to 'top-up' their vehicle when they reach the location, rather than or as well as, find the sites nearest to them at the start of a journey.

A successful match to the user's request, results in a response containing fuel price and associated site data for up to 5 sites (for a 2 or 5 mile radius from the location) or up to 10 sites (for a 10 mile radius from the location). The response data for each site also includes a ranking field which can either be related to price or distance, depending on the user's specified interest. The prices returned in the response data are either next-day or in-day prices, dependent on subscription.

The bespoke application can extract, validate, and consume the response data. In its simplest form, the data could be used to create and present a list in ranked order showing the name and location of each site and the current price for the requested type of fuel (petrol or diesel). More sophisticated uses might be to include directions to each site from the specified location; add a pointer on a map; add a link to a Google StreetView showing an image of the forecourt to allow it to be easily identified from the road; and allow the user to view and add review comments about a site and the facilities available.

Clients who want to use the API, must first apply to obtain a Neartime Service account. Clients need to choose whether they want next day or in-day pricing. All variables returned in the response are bundled into an asset, and given a unique ID known as an asset ID.

Refer to the process flow diagram below.



- 1** To use the API, you must login in using your username and password.
- 2** For a successful log in, the API sends an authenticated token which you can then use for search requests. Each token has a lifetime of up to 30 minutes, after which you must send a subsequent log in request.
- 3** Once logged in, you can query the API by supplying your SSO ID (Single Sign On ID), token, client ID; the asset ID identifying the Catalist dataset and the search data. The search data comprises the required location (longitude and latitude or postcode); the size of the distance ring from the location (2, 5 or 10 miles); the grade or type of fuel (petrol or diesel); and the required metric which is used to rank the results (price or distance).
- 4** The API returns the details of all sites that match the criteria specified in the search request. For a 2 or 5 mile distance ring, the details of up to 5 sites are returned in the response and for a 10 mile distance ring, the details of up to 10 sites are returned. The results for each site include a ranking field which is related to price or distance as given the rank metric specified in the request.
- 5** The values of the matching site data are extracted from the response and used within your bespoke application to create the required output, e.g. a display of the details of petrol stations nearest or cheapest to the specified location.

1.3 Purpose of this document

This document provides technical information about:

- how to get an API account;
- the log in request and response, including examples;
- the two types of search request (GET and POST) and their responses, including examples;
- the various HTTP responses that may be received from the API. For error responses this includes possible causes and their resolution.

1.4 Document audience and pre-requisites

This document is intended to be used by developers wanting to use the API in their own bespoke mobile or Web-based applications.

There are many ways to create the HTTP GET and POST requests needed to log in and query the API and to consume the responses. This can be from simple browser development tools and add-ins; tools and libraries such as cURL and JQuery; to development environments for programming languages such C# and Java which provide libraries that include creating instances of an HTTP client or Web request.

It follows, therefore, that any API guide cannot be too prescriptive and can only give the 'bare bones' of information about the API. For this reason, the request examples used in this guide are in the Windows version of cURL. However, sufficient information is given about the construction of each HTTP request to allow you to build them using any preferred language or tool.

It is expected that you have a good understanding of:

- HTTP terminology and the general construct of GET and POST requests, including standard and custom headers, GET query strings, and HTTP responses.
- The JSON (JavaScript Object Notation) format and syntax and the notion of key and value pairs.
- The cURL (Client URL) library, its installation, and its command line syntax. The examples in this guide uses the Windows version of cURL, but of course there are other versions including those for Linux and Python (urllib2).

1.5 Getting a Catalist API account

1.5.1 Neartime domain names

You can ask to be registered to two Neartime Service domains:

- stg.neartime.experian.co.uk – this is the domain name for the Stage environment. You can use the Stage environment for testing purposes.
- neartime.experian.co.uk– this is domain name for the Live environment. When you have finished testing your application, you can switch to the URL for the Live environment.

1.5.2 Talking with your Experian Account Manager

If you want to use the API you must first talk with your Experian Account Manager. The conversation will include finding out whether you want to receive next-day or in-day pricing.

1.5.3 Setting up your account

Registering the Neartime Service

Your Experian Account Manager will apply to the Experian Helpdesk and ask them to set up your account on the Experian Plexus Administration Site and give you access to the Neartime Service which facilitates the Catalist API.

The Plexus Administration Site is used for other services and applications such as Location Analyst, iCoder Online, Mosaic Audience and the Segmentation Portal. If you already use one of these services or sites, you will already have a username and password. In which case, the Helpdesk will only need to add the Neartime Service to your account and arrange for the access to the Catalist API dataset to be set up.

You can manage your password through any of the Plexus services or applications to which you have access, i.e. the same username and password is used for all services and sites to which you subscribe.

Any questions you have about your account should be sent to the Experian Helpdesk (EMSUKHelpdesk@experian.com).

Soon after the creation or update of your account you will receive an automated registration e-mail giving your username and password (if a new account has been created).

Getting access to the Catalist dataset

Giving you access to the Neartime Service is just part of the onboarding process. You will also need to be given access to the Catalist dataset on each of the two domains described in Subsection 1.5.1. The requests you make to the API, require you to specify an asset ID which identifies the dataset and pricing option to which you have access and licensed.

You will be notified by e-mail when this is complete and be given a 5-digit client ID and the asset ID for your Catalist data. You can then access the API.

Client IDs are used by the Neartime Service to match your username to a list of authorised usernames associated with that ID. For instance, you may have different users within your company that can access the API. The client ID is authorised to send a lookup request for an asset, but the requestor (username) is also logged.

2 Logging in

2.1 About the log in request

To be able to perform a search of the Catalist database, you must first log in to the API to authenticate the session and receive a token. You must send your username and password, in JSON format, in the body of the request.

Currently, you are only allowed one token at any one time which is valid for 30 minutes. If you make any subsequent log in attempts, this invalidates the token of any active session you may have started and causes a new token to be created.

2.1.1 HTTP summary details

The table below summarises the HTTP details of the log in request:

HTTP Element	Value
Request method:	POST
Standard header:	Content-Type:application/json
URL or endpoint:	https://[domain_name]/catalist/login
Request body:	<pre>{ "username": "USERNAME_VALUE", "password": "PASSWORD_VALUE" }</pre>

Note: See Subsection 1.5.1 for details of the domain names you can use for the [domain_name] placeholder.

2.1.2 Standard header

For the log in request, you must set the value of the standard Content-Type header to a MIME type of “application/json”, i.e. Content-Type:application/json.

2.1.3 Request body key and value pairs

The two key and string value pairs in the log in request body (in JSON format) given above are defined as follows:

- "username": "USERNAME_VALUE" – set the value to be the same as the username given in the registration e-mail e.g.
"username": "client.username@client.com"
- "password": "PASSWORD_VALUE" – set the value to be the same as your current Plexus password, e.g. "password": "client_pAssw0rd".

2.1.4 Example log in request

The following is a simple Windows cURL command line example which is used to demonstrate the content of a log in request. See section 1.5.1 for details of the domain names.

```
curl -X POST -H "Content-Type:application/json"
"https://[domain_name]/catalist/login -d
"{\"username\":\"client.username@client.com\", \"password\":\"
client_pAssw0rd\"}"
```

2.2 Log in response

2.2.1 Successful log in

If the log in request is successful, the API sends a 200 (OK) response. The body of the response is a single JSON key and string value pair. The value of the `token` key is a hexadecimal string. An example response is shown below:

```
{ "token": "713dbcaa-21f5-4124-a576-176805b7b326" }
```

2.2.2 Error response

If the log in request is unsuccessful, the API sends a different response.

The server may send a 429 (Too Many Requests) response to limit the loading on the network. If this occurs you will need to wait a specified amount of time dependent on the endpoint before retrying. The time remaining (in milliseconds) is included in the response message.

See Section 4 for information about the other types of response which you may receive for a log in request, what they mean, and how to resolve the issue.

2.2.3 Processing the log in response

In a real-world Web-based or desktop application, if the log in request is successful, the token is likely to be programmatically extracted from the JSON response and then passed through, or made accessible, or to a search request (see Section 3).

It is also expected that you would write application code to handle any response which is not successful.

3 Sending and processing searches

3.1 Types of search request

There are two types of request that you can use to initiate a search of the Catalist database:

- a GET request, where you supply the search data in an HTTP query string.
- a POST request, where you supply the search data in a JSON request body.

3.2 Using a GET search request

3.2.1 About the request

You can use a GET request to perform a search of the Catalist database by sending the search data as parameter names and values in an HTTP query string.

To be accepted as a valid request, any spaces in the query string values must be replaced by either a plus character (+) or %20. You can achieve this by passing the complete URL through a URL encoding function before making the request.

You must authenticate the query by including two custom headers to the request: one containing your username, and the other a valid token received from a log in request.

3.2.2 HTTP summary details

The table below summarises the details of the request:

HTTP Element	Value
Request method:	GET
Custom headers:	username=USERNAME_VALUE token=TOKEN_VALUE
URL or endpoint (with query string):	For a search which uses a longitude and latitude to define the required location: <pre>https://[domain_name]/catalist/search?clientId=CLIENTID_VALUE&assetId=ASSETID_VALUE&longitude=LONGITUDE_VALUE&latitude=LATITUDE_VALUE&drivedistance=DRIVEDISTANCE_VALUE&grade=GRADE_VALUE&rankmetric=RANKMETRIC_VALUE</pre> -or- for a search which uses a postcode to define the required location: <pre>https://[domain_name]/catalist/search?clientId=CLIENTID_VALUE&assetId=ASSETID_VALUE&postcode=POSTCODE_VALUE&drivedistance=DRIVEDISTANCE_VALUE&grade=GRADE_VALUE&rankmetric=RANKMETRIC_VALUE</pre>
Note: See Subsection 1.5.1 for details of the domain names you can use for the [domain_name] placeholder.	

3.2.3 Custom headers

You must set two custom headers in the GET request:

- `username=USERNAME_VALUE` – set the value to be the same as the username given in your registration e-mail and used at log in e.g.
`username=client.username@client.com`.
- `token=TOKEN_VALUE` – set the value to be the same as the token you received in the log in response, e.g. `token=713dbcaa-21f5-4124-a576-176805b7b326`.

3.2.4 Query string parameter name and value pairs

The parameter name and value pairs in the URL's query string given above are defined as follows:

- `clientId=CLIENTID_VALUE` – set the value to be the same as the client ID which you received in the registration email, e.g. `clientId=12345`.
- `assetId=ASSETID_VALUE` – set the value to be the same as the asset ID which you received in the registration email, e.g. `assetId=88883A`.
- `latitude=LATITUDE_VALUE` – if used, set the value to the latitude of the required location in WGS format and up to 6 decimal places, e.g. `latitude=52.951138`. If you want to specify the location using a postcode, then this parameter must not be supplied in the request.
- `postcode=POSTCODE_VALUE` – if used, set the value to a UK postcode for the required location, e.g. for the UK `postcode=NG13+8EH`. It should be noted that the space in the value of example `postcode` parameter has been replaced by a plus (+) character. If an unencoded space is sent instead, the API will send an HTTP 505 (HTTP Version Not Supported) response. If you want to specify the location using a longitude and latitude, then this parameter must not be supplied in the request.
- `drivedistance=DRIVEDISTANCE_VALUE` – set the value to the required distance from the location, i.e. 2, 5 or 10 miles, e.g. `drivedistance=5`. A value of 2 or 5 will return the details of up to 5 sites in the response, while a value of 10 will return the details of up to 10 sites.
- `grade=GRADE_VALUE` – set the value to type or grade of fuel to do with the required price data, i.e. `petrol` (for standard unleaded petrol), `diesel` (for standard diesel), `super-unleaded` (for supergrade unleaded petrol) or `super-diesel` (for supergrade diesel), e.g. `grade=petrol`.
- `rankmetric=RANKMETRIC_VALUE` – set the value to how you want the results to be ranked, i.e. `distance` or `price`, e.g. `rankmetric=price`. A value of 1 for the `rank` key in the response identifies the cheapest or nearest petrol station to the specified location.

3.2.5 Example GET search requests

The following is a simple Windows cURL command line example which is used to demonstrate the content of a GET request which uses a longitude and latitude to define the location for the search of the Catalist database. See section 1.5.1 for details of the domain names.

```
curl -H "username:client.username@client.com" -H  
"token:713dbcaa-21f5-4124-a576-176805b7b326"  
"https://[domain_name]/catalist/search?clientID=12345&assetID=  
88883A&longitude=-0.936500&latitude=52.951138&drivedistance=5  
&grade=diesel&rankmetric=price"
```

The following is a simple Windows cURL command line example which is used to demonstrate the content of a GET request which uses a UK postcode to define the location for the search of the Catalist database. See section 1.5.1 for details of the domain names.

```
curl -H username:client.username@client.com" -H  
"token:713dbcaa-21f5-4124-a576-176805b7b326"  
"https://[domain_name]/catalist/search?clientID=12345&assetID=  
88883A&postcode=NG13+8EH&drivedistance=10&grade=petrol  
&rankmetric=distance"
```

It should be noted that the space in the value of example `postcode` parameter has been replaced by a plus (+) character. If an unencoded space is sent instead, the API will send an HTTP 505 (HTTP Version Not Supported) response.

3.3 Using a POST search request

3.3.1 About the request

You can use a POST request to perform a search of the Catalist database by sending your username, a valid token, the search data in JSON format, in the body of the request.

3.3.2 HTTP summary details

The table below summarises the details of the request:

HTTP Element	Value
Request method:	POST
Standard header:	Content-Type:application/json
URL or endpoint:	https://[domain_name]/catalist/search
Request body:	For a search which uses a longitude and latitude to define the required location: <pre>{ "username": "USERNAME_VALUE", "token": "TOKEN_VALUE", "clientId": "CLIENTID_VALUE", "assetId": "ASSETID_VALUE", "longitude": "LONGITUDE_VALUE", "latitude": "LATITUDE_VALUE", "drivedistance": "DRIVEDISTANCE_VALUE", "grade": "GRADE_VALUE", "rankmetric": "RANKMETRIC_VALUE" }</pre> -or- for a search which uses a postcode to define the required location: <pre>{ "username": "USERNAME_VALUE", "token": "TOKEN_VALUE", "clientId": "CLIENTID_VALUE", "assetId": "ASSETID_VALUE", "postcode": "POSTCODE_VALUE", "drivedistance": "DRIVEDISTANCE_VALUE", "grade": "GRADE_VALUE", "rankmetric": "RANKMETRIC_VALUE" }</pre>

Note: See Subsection 1.5.1 for details of the domain names you can use for the [domain_name] placeholder.

3.3.3 Standard header

For a POST request, you must set the value of a standard Content-Type header to a MIME type of “application/json”, i.e. Content-Type:application/json.

3.3.4 Request body key and value pairs

The key and value pairs in the request body given above are defined as follows:

- "username": "USERNAME_VALUE" – set value to be the same as the username given in the registration e-mail and used at log in e.g. "username": "client.username@client.com".
- "token": "TOKEN_VALUE" – set the value to be the same as the token you received in the log in response, e.g. "token": "713dbcaa-21f5-4124-a576-176805b7b326".
- "clientId": "CLIENTID_VALUE" – set the value to be the same as the client ID which you received in the registration email, e.g. "clientId": "12345".

- `"assetId": "ASSETID_VALUE"` – set the value to be the same as the asset ID which you received in the registration email, e.g. `"assetId": "88883A"`.
- `"longitude": "LONGITUDE_VALUE"` – if used, set the value to the longitude of the required location in WGS format and up to 6 decimal places, e.g. `"longitude": "-0.936500"`. If you want to specify the location using a postcode, then this key must not be supplied in the request.
- `"latitude": "LATITUDE_VALUE"` – if used, set the value to the latitude of the required location in WGS format and up to 6 decimal places, e.g. `"latitude": "52.951138"`. If you want to specify the location using a postcode, then this key must not be supplied in the request.
- `"postcode": "POSTCODE_VALUE"` – if used, set the value to a UK postcode for the required location, e.g. for the UK `"postcode": "NG13 8EH"`. If you want to specify the location using a longitude and latitude, then this key must not be supplied in the request.
- `"drivedistance": DRIVEDISTANCE_VALUE` – set the integer value to the required distance from the location, i.e. 2, 5 or 10 miles, e.g. `"drivedistance": 5`. A value of 2 or 5 will return the details of up to 5 sites in the response, while a value of 10 will return the details of up to 10 sites.
- `"grade": "GRADE_VALUE"` – set the value to the type or grade of fuel to do with the required price data, i.e. petrol (for standard unleaded petrol), diesel (for standard diesel), super-unleaded (for supergrade unleaded petrol) or super-diesel (for supergrade diesel), e.g. `grade=petrol`.
- `"rankmetric": "RANKMETRIC_VALUE"` – set the value to how you want the results to be ranked, i.e. distance or price, e.g. `"rankmetric": "price"`. A value of 1 for the rank key in the response identifies the cheapest or nearest petrol station to the specified location.

3.3.5 Example POST search requests

The following is a simple Windows cURL command line example which is used to demonstrate the content of a POST request which uses a longitude and latitude to define the location for the search of the Catalist database:

```
curl -X POST -H "Content-Type:application/json"
https://[domain_name]/catalist/search -d '{"username\":"
"client.username@client.com\","token\":"13083fcf-cdf5-42de-
88bc-6c0bc1022ecb\","clientId\":"12345\","
"assetId\":"88883A\","longitude\":"1.100001\","
"latitude\":"50.100001\","drivedistance\":"10,
"grade\":"petrol\","rankmetric\":"price"}'
```

The following is a simple Windows cURL command line example which is used to demonstrate the content of a POST request which uses a UK postcode to define the location for the search of the Catalist database:

```
curl -X POST -H "Content-Type:application/json"
https://[domain_name]/catalist/search -d "{\"username\":
\"client.username@client.com\", \"token\":\"13083fcf-cdf5-42de-
88bc-6c0bc1022ecb\", \"clientId\":\"12345\",
\"assetId\":\"88883A\", \"postcode\":\"NG13 8EH\",
\"drivedistance\":10, \"grade\":\"petrol\",
\"rankmetric\":\"price\"}"
```

3.4 Search response (GET and POST)

3.4.1 Matched search

If there has been a successful match with the search data you have supplied, the API sends a 200 (OK) response. The body of the response is in JSON format. The first key and value is a count of exactly how many sites have been returned. The `count` key is followed by a `sites` array comprising the details of each matching site in rank order from 1 to the number sites in the array.

An example of a successful match is shown below. For the sake of brevity, this shows showing 2 out of 6 sites returned in the response, ranked first and last, by the cheapest price for petrol:

```
{
  "count":6,
  "sites":[
    {
      "brand": "JET",
      "catnumber": "9735",
      "county": "NOTTINGHAMSHIRE",
      "date": "2021-01-23",
      "grade": "petrol",
      "latitude": 52.912565,
      "longitude": -1.043387,
      "postcode": "NG12 3HQ",
      "price": 1219,
      "rank": 1,
      "sitename": "COTGRAVE SERVICE STATION",
      "street": "MAIN ROAD",
      "town": "NOTTINGHAM"
    },
    ...
    {
      "brand": "SHELL",
      "catnumber": "4221",
      "county": "NOTTINGHAMSHIRE",
      "date": "2021-01-25",
```

```

        "grade": "petrol",
        "latitude": 52.949205,
        "longitude": -0.979218,
        "postcode": "NG13 8AY",
        "price": 1239,
        "rank": 8,
        "sitename": "SHELL SAXONDALE",
        "street": "SAXONDALE CROSSROADS",
        "town": "NOTTINGHAM"
    }
]
}

```

The table gives the details about each key in the response.

Key Name	Data Type	Description
count	Integer	The number of sites that have been returned in the <code>sites</code> array. If a <code>drivedistance</code> of 2 or 5 was specified in the request, this will have a value of between 1 and 5. If a <code>drivedistance</code> of 10 was specified in the request, this will have a value of between 1 and 10.
sites > brand	String	The brand of the site as perceived by the customer and on the pole, e.g. "BP". Where there is no site branding, or it is inconsistent, "UNBRANDED" is used.
sites > catnumber	String	This is the unique Experian Catalist reference number for the site. Numbers for sites in the UK and the Isle of Man start at 1.
sites > county	String	The name of the county in which the site is located, e.g. "OXFORDSHIRE".
sites > date	String	The date that the price data was supplied in CCYY-MM-DD format, e.g. "2019-11-25". See Note below.
sites > grade	String	The type or grade of fuel sold at the site and specified in the request, i.e. "petrol" (for standard unleaded petrol), "diesel" (for standard diesel), "super-unleaded" (for supergrade unleaded petrol) or "super-diesel" (for supergrade diesel). See Note below.
sites > latitude	String	The latitude of the site's location in WGS84 format and up to 6 decimal places, e.g. "50.804466".

Key Name	Data Type	Description
<code>sites > longitude</code>	String	The longitude of the site's location in WGS84 format and up to 6 decimal places, e.g. "-0.512047".
<code>sites > postcode</code>	String	The postcode for a UK site, e.g. "RG5 5DB".
<code>sites > price</code>	Integer	The next-day or in-day modal price/litre of the fuel to the nearest tenth of a penny for the <code>date</code> specified. For example, a value of 1217 represents 121.7p/litre. See Note below.
<code>sites > rank</code>	Integer	The rank position of this site in relation of all other sites returned in the response and based on the value of the <code>rankmetric</code> parameter or key specified in the request: "price" or "distance". A value of 1 represents the highest ranking site, i.e. the cheapest site or the nearest site to the location specified in the request.
<code>sites > sitename</code>	String	The name of the site, as it is commonly recognised, e.g. "SHELL PINE". This is not the name of the company operating the site.
<code>sites > street</code>	String	The name of the street that provides direct access to the site, e.g. "EDGEWARE ROAD". If the site is on a corner, with direct access from more than one street, the primary street is the busier of the two. A cardinal point may also be used at the end of the address line for some sites which straddle a motorway or trunk road e.g. "M1 NORTHBOUND".
<code>sites > town</code>	String	The name of postal town or city near to or in which the site is located, e.g. "HENLEY-ON-THAMES".

Note: If there is no price data for a site or it is more than 10 days old, the `date`, `grade` and `price` fields are not included in the response for that site.

3.4.2 Non-matched search

If the search request results in a non-match, a 200 (OK) response is given but with empty JSON curly braces, i.e. `{ }`.

3.4.3 Error response

If there a problem occurred processing the request the server will give a different and HTTP response.

The server may send a 429 (Too Many Requests) response to limit the loading on the network. If this occurs you will need to wait a specified amount of time dependent on the endpoint before retrying. The time remaining (in milliseconds) is included in the response message.

See Section 4 for information on the other types of response which may be received for a request, what they mean, and how to resolve the issue.

3.4.4 Processing the response data

If the search request results in a successful match, then it is likely that you will write code to extract, validate and consume the response data. In its simplest form, the data could be used to create and present a list in ranked order showing the name and location of each site and the current price for the requested type of fuel (petrol or diesel). More sophisticated uses might be to include directions to each site from the specified location; add a pointer on a map; add a link to a Google StreetView showing an image of the forecourt to allow it to be easily identified from the road; and allow the user to view and add review comments about a site and the facilities available.

Note: With the exception of `sites > grade`, all alphabetic characters returned as key values in the response data are in uppercase characters, e.g. a `street` key value of `"EDGEWARE ROAD"`. Depending on if and how you want to display any text to the user, you may want to process the text that is returned so that is in proper case, i.e. the first letter of each word is a capital, e.g. `"Edgeware Road"`.

It is also expected that you would write application code to handle any response where no match is found (`{ }` given) or which indicates that there was a problem (see Section 4 for details).

4 HTTP responses

4.1 Introduction

The following provides the details of the possible responses to the log in request, and search requests (GET and POST). In some cases, the text of the response is the standard HTML response given by the server, while in other cases a custom response has been created to be used specifically with the API.

The details given provide the meaning of the response, and if the response is an error, the possible causes and the resolution.

The names in parentheses are the standard HTTP names for the responses.

4.2 200 (OK) responses

4.2.1 200 – Matched search

This response is given to a search request where there has been a successful match with the search data supplied.

The body of the response is in JSON format. The first key and value is a count of exactly how many sites have been returned. This is followed by a `sites` array comprising the details of each matching site in rank order from 1 to the number sites in the array. For example, for a single matching site:

```
{
  "count":1,
  "sites":[
    {
      "brand": "JET",
      "catnumber": "9735",
      "county": "NOTTINGHAMSHIRE",
      "date": "2021-01-23",
      "grade": "petrol",
      "latitude": 52.912565,
      "longitude": -1.043387,
      "postcode": "NG12 3HQ",
      "price": 1219,
      "rank": 1,
      "sitename": "COTGRAVE SERVICE STATION",
      "street": "MAIN ROAD",
      "town": "NOTTINGHAM"
    }
  ]
}
```

4.2.2 200 – Non-matched search

This response may be given to a request where a search of the Catalist database resulted in no match.

It includes a standard `Content-Length` header set to 2 and the body with empty JSON curly braces, i.e. `{ }`.

4.2.3 200 – Successful log in

This response may be given to a successful log in request.

The body of the response contains a single JSON key and value pair comprising the `token` key name and a hexadecimal string e.g.

```
{ "token": "713dbcaa-21f5-4124-a576-176805b7b326" }
```

4.3 400 – (Bad Request) responses

4.3.1 400 – Invalid value for '[name]'.

This is a custom response given by the server if it detects a problem with the value of a parameter or key in a search request (GET or POST).

This message may be displayed for the `latitude`, `longitude` and `postcode` parameters or keys.

The body of the custom response contains two JSON key and value pairs:

```
{ "code": 400, "response": "Invalid value for `[name]`." }
```

Cause or Resolution	Description
Cause:	A problem was detected in the search request with the value of the parameter or key identified in the <code>response</code> key of the message. This could be because the value is mis-spelt or incorrectly formatted.
Resolution:	Check that the parameter or key identified in the <code>response</code> key of the message is valid.

4.3.2 400 – Invalid value for '[name]', must be [valid_values]

This is a custom response given by the server if it detects a problem with the value of a parameter or key in a search request (GET or POST).

This message may be displayed for the `drivedistance`, `grade` and `rankmetric` parameters or keys. The parameter or key and possible valid values are given in the `response` key of the message.

The body of the custom response contains two JSON key and value pairs:

```
{  "code": 400,  "response": "Invalid value for `[name]`, must be [valid_values]."}
```

Cause or Resolution	Description
Cause:	A problem was detected in the search request with the value of the parameter or key identified in the <code>response</code> key. This could be because the value is mis-spelt.
Resolution:	Check that the parameter or key value matches one of the valid values identified in the <code>response</code> key of the message.

4.3.3 400 – 'latitude' is out of range, must be between -90 and 90.

This is a custom response given by the server if it detects that the value of the `latitude` parameter or key in a search request (GET or POST) is out of range.

The body of the custom response contains two JSON key and value pairs:

```
{  "code": 400,
  "response": "'latitude' is out of range, must be between -90 and 90."
}
```

Cause or Resolution	Description
Cause:	The value of the <code>latitude</code> parameter or key is outside of the range -90 and 90.
Resolution:	Ensure the value of the <code>latitude</code> parameter or key is within range. Geographically, a latitude can have any value between -90 and 90. However, in the context of the UK and IoM, (the area covered by the Catalist API) a valid latitude should be between from 49.9 and 60.84.

4.3.4 400 – `longitude` is out of range, must be between -180 and 180.

This is a custom response given by the server if it detects that the value of the `longitude` parameter or key in a search request (GET or POST) is out of range.

The body of the custom response contains two JSON key and value pairs:

```
{  "code": 400,
   "response": "`longitude` is out of range, must be between -
180 and 180."
}
```

Cause or Resolution	Description
Cause:	The value of the <code>longitude</code> parameter or key is outside of the range -180 and 180.
Resolution:	Ensure the value of the <code>longitude</code> parameter or key is within range. Geographically, a longitude can have any value between -180 and 180. However, in the context of the UK and IoM, (the area covered by the Catalist API) a valid longitude should be between -8.62 and 1.77.

4.3.5 400 – Missing query parameter `[name]`

This is a custom response given by the server if it detects a problem with a parameter in a GET search request.

The body of the custom response contains four JSON key and value pairs:

```
{
  "detail": "Missing query parameter '[name]'",
  "status": 400,
  "title": "Bad Request",
  "type": "about:blank"
}
```

Cause or Resolution	Description
Cause:	A problem was detected with the parameter identified in the <code>detail</code> key. This could be because the parameter is missing from the request or that it has been mis-spelt.
Resolution:	Check that the parameter is supplied in the request and that it is spelt correctly.

4.3.6 400 – '[name]' is a required property

This is a custom response given by the server if it detects a problem with a key in a POST request (log in or query).

The body of the custom response contains four JSON key and value pairs:

```
{  "detail": "'username' is a required property",  "status": 400,  "title": "Bad Request",  "type": "about:blank"}
```

Cause or Resolution	Description
Cause:	A problem was detected with the key identified in the <code>detail</code> key. This could be because the key is missing from the request or that it has been mis-spelt.
Resolution:	Check that the key is supplied in the request and that it is spelt correctly.

4.3.7 400 – Request body is not valid JSON

This is a custom response given by the server if it detects there is a problem with the formatting of the JSON content in the body of a POST request (log in or query).

The body of the custom response contains four JSON key and value pairs:

```
{  "detail": "Request body is not valid JSON",  "status": 400,  "title": "Bad Request",  "type": "about:blank"}
```

Cause or Resolution	Description
Cause:	A problem was detected with the formatting of the JSON content in the body of the POST request.
Resolution:	Check that the formatting of the JSON content in the POST request and correct the formatting error.

4.3.8 400 – Request can not include both postcode and latitude/longitude parameters

This is a custom response given by the server if the request included the postcode and the latitude/longitude parameters or keys.

The body of the custom response contains two JSON key and value pairs:

```
{ "code": 400, "response": "Request can not include both
postcode and latitude/longitude parameters."}
```

Cause or Resolution	Description
Cause:	The request includes the postcode and the latitude/longitude parameters or keys.
Resolution:	Remove the postcode or the latitude/longitude parameters or keys, as appropriate, from the request and retry.

4.3.9 400 – Request must include either postcode or latitude/longitude parameters

This is a custom response given by the server if the request does not include the postcode or the latitude/longitude parameters or keys.

The body of the custom response contains two JSON key and value pairs:

```
{ "code": 400, "response": "Request must include either
postcode or latitude/longitude parameters."}
```

Cause or Resolution	Description
Cause:	The request does not include the postcode or the latitude/longitude parameters or keys.
Resolution:	Add the postcode or the latitude/longitude parameters or keys, as appropriate, to the request and retry.

4.3.10 400 – '[value]' is not of type '[data_type]'

This is a custom response given by the server if it detects a data type mismatch with the value of a key in a log in or a POST search request.

The body of the custom response contains four JSON key and value pairs:

```
{  "detail": "'[value]' is not of type '[data_type]'",
  "status": 400,  "title": "Bad Request",  "type": "about:blank"}
```

Cause or Resolution	Description
Cause:	A problem was detected with the data type of a key value identified in the <code>detail</code> key in the request.
Resolution:	For a key value which should be a string, ensure that the value is enclosed by double quotes.
	For a key value which should be an integer, ensure that the value is not enclosed by double quotes.

4.4 401 – (Unauthorized) responses

4.4.1 401 – Asset Not Found.

This is a custom response given by the server if it detects that the value of the `assetId` supplied in the search request does not exist. The body of the custom response contains two JSON key and string value pairs:

```
{"code":401,"response":"Asset Not Found."}
```

Cause or Resolution	Description
Cause:	The value <code>assetId</code> given in the search request could not be found by the server.
Resolution:	Check the value of the <code>assetId</code> matches one of those given in the registration e-mail, and correct if necessary.
	If the value of the <code>assetId</code> does match the one given in the registration e-mail, contact the Helpdesk to report the problem and receive the correct asset ID.

4.4.2 401 – Invalid token

This is a custom response given by the server if it detects a problem with the token supplied in the search request.

The body of the custom response contains a two JSON key and value pairs:

```
{"code":401,"response":"Invalid token"}
```

Cause or Resolution	Description
Cause:	A problem was detected with the value of the <code>token</code> header or key. The most likely cause of this is because the token used in the search request is more than 30 minutes old. It may also be caused if the token in the query is different to the one received from the log in request.
Resolution:	If the token is not more than 30 minutes old, check that the value used in the search request is the same as the one given in the log in request. If the token is more than 30 minutes old, log in to the API to obtain a new token.

4.4.3 401 – Unauthorized Client

This is a custom response given by the server if it detects a problem with the credentials given in a query request. The body of the custom response contains two JSON key and string value pairs:

```
{"code":401,"response":"Unauthorized Client"}
```

Cause or Resolution	Description
Cause:	A problem was detected with the credentials in a query request where the value of the <code>ssoId</code> or <code>clientId</code> key may be incorrect.
Resolution:	Check the value of the <code>ssoId</code> matches the username in the registration e-mail. The value of the <code>clientId</code> key must match the value given in the registration e-mail to the client. Check the value of the key against the one in the e-mail. If the response is still given, contact the Helpdesk.

4.4.4 401 – Unauthorized Client, null token

This is a custom response given by the server if it detects a problem with the username and/or the password given in log in request, so no token was returned.

The body of the custom response contains a two JSON key and value pairs:

```
{"code":401,"response":"Unauthorized Client, null token"}
```

Cause or Resolution	Description
Cause:	A problem was detected with the credentials given in a log in request where the value of the <code>password</code> or the <code>username</code> key may be incorrect.
Resolution:	The value of the <code>username</code> key must match that given by the username in the registration e-mail. The value and case of <code>password</code> key must match the current password which is used to log in to Plexus applications. Check the value of both keys and correct as necessary. Ensure that value of the <code>password</code> key matches the latest password, particularly if it was recently changed by someone else in your organisation.

4.5 404 – (Not Found) response

This is a standard response given by the server if it does not recognise the endpoint or URL supplied in a log in request or search request.

The response body is supplied as an HTML encoded page.

Cause or Resolution	Description
Cause:	The URL or endpoint specified in a log in request or search request was not matched to any resource by the server and so a standard error was returned.
Resolution:	Check the spelling of the log in request URL or the search request URL and confirm that it matches the ones specified in this document, i.e. for a log in request: <pre>https://[domain_name]/catalist/login</pre> or for a search request: <pre>https://[domain_name]/catalist/search</pre>

4.6 415 – (Unsupported Media Type) response

This is a custom response given by the server if it detects a problem with the media type which has been set in the standard `Content-Type` header of a log in or POST search request.

The body of the custom response contains four JSON key and value pairs:

```
{  "detail": "Invalid Content-type (application/x-www-form-urlencoded), expected JSON data",  "status": 415,  "title": "Unsupported Media Type",  "type": "about:blank"}
```

Cause or Resolution	Description
Cause:	The MIME type in the standard <code>Content-Type</code> header for the POST request has been set incorrectly or may contain a typographical error.
Resolution:	Ensure the value of the standard <code>Content-Type</code> header is set to a MIME type of <code>"application/json"</code> , i.e. <code>Content-Type:application/json</code> .

4.7 429 – (Too Many Requests) response

This is a custom response given to limit the loading on the network. The response may be given for any type of request. If this occurs you will need to wait a specified amount of time dependent on the endpoint before retrying.

The body of the custom response contains three JSON key and value pairs:

```
{  "code":429,  "response":"Limit Reached for [endpoint], try again in [time_ms]",  "remaining":[time_ms]}
```

Cause or Resolution	Description
Cause:	The server has detected that the loading on the network is too high.
Resolution:	You should write code to deal with this response, and waiting the specified time remaining before retrying the request.

4.8 500 – (Internal Server Error) responses

4.8.1 500 – Internal Server Error

This is a custom response given by the server if an unexpected and severe problem occurs.

The body of the custom response contains four JSON key and value pairs:

```
{ "detail": "The server encountered an internal error and was
unable to complete your request. Either the server is
overloaded or there is an error in the application.", "status":
500, "title": "Internal Server Error", "type": "about:blank"}
```

Cause or Resolution	Description
Cause:	The cause of these errors are unknown, but the system is closely monitored by Experian for any of these failures.
Resolution:	Contact the Helpdesk to report this problem for further investigation.

4.8.2 500 – Problem with authentication

This is a custom response given by the server if there was an issue with a login request. The body of the custom response contains two key and value pairs:

```
{"code":500,"response":"Problem with authentication"}
```

Cause or Resolution	Description
Cause:	There was an issue with logging in because of a communication issue with the Plexus Administration server.
Resolution:	Try again. If the issue is still present, contact the Helpdesk to report this problem for further investigation.

4.8.3 500 – Problem with login

This is a custom response given by the server if there was an issue retrieving a token from the Plexus Administration server. The body of the custom response contains two key and value pairs:

```
{"code":500,"response":"Problem with login"}
```

Cause or Resolution	Description
Cause:	There was an issue retrieving a token from the Plexus Administration server.
Resolution:	Try again. If the issue is still present, contact the Helpdesk to report this problem for further investigation.

4.9 500 – (Internal Server Error) response

This is a custom response given by the server if an unexpected and severe problem occurs.

The body of the custom response contains four JSON key and value pairs:

```
{ "detail": "The server encountered an internal error and was
unable to complete your request. Either the server is
overloaded or there is an error in the application.", "status":
500, "title": "Internal Server Error", "type": "about:blank"}
```

Cause or Resolution	Description
Cause:	The cause of these errors are unknown, but the system is closely monitored by Experian for any of these failures.
Resolution:	Contact the Helpdesk to report this problem for further investigation.

4.10 503 – (Service Unavailable) responses

4.10.1 503 – Internal refresh in progress

This is a custom response given by the server if a refresh was in progress when the request was made.

The body of the custom response contains a two JSON key and value pairs:

```
{ "code": 503, "response": "Internal refresh in progress" }
```

Cause or Resolution	Description
Cause:	There was temporary problem with the server in the cluster which received the request.
Resolution:	Try the request again as it is likely to succeed when the refresh operation has been completed. If the request still fails, try again in 10 minutes and if it still fails report the problem to the Helpdesk.

4.10.2 503 – Server error

This is a custom response given by the server if a serious server error occurs when a request is made.

The body of the custom response contains a two JSON key and value pairs:

```
{"code":503,"response":"Server error"}
```

Cause or Resolution	Description
Cause:	The cause of these errors are unknown, but the system is closely monitored by Experian for any of these failures.
Resolution:	Report the problem to the Helpdesk, and the status of a fix.

4.11 505 – (HTTP Version Not Supported) response

This is a standard response given by the server which indicates that the version of HTTP used in the request is not handled by the server.

The response body is supplied as an HTML encoded page.

Causes or Resolution	Description
Causes:	There are several causes for this response: - the protocol has been explicitly defined in the request, which has not been understood by the server. This includes specifying an invalid version number or including one or more trailing spaces in the version specified. - raw (unencoded) spaces may be present in the query string values of a GET request. - the values of standard headers in the request may not be legal for HTTP version 1.1.
Resolution:	Check that if the protocol has been explicitly defined in the request it is HTTP version 1.1 and it has been correctly defined. Check that and spaces in the values the query string in a GET request are properly encoded as plus (+) characters or %20. Check the values for any standard headers in the request are valid and are supported by HTTP version 1.1. If the problem continues, contact the Helpdesk.