Quickstart Guide for GraphQL API in DeltaV™ SaaS SCADA

Zedi is Now DeltaV SaaS SCADA

GraphQL API Tools to Use

GraphIQL Feen
DeltaV SaaS SCADA recommends using the GraphIQL Feen, Chrome extension, or similar, to fully explore the GraphQL API available for integration.

https://chrome.google.com/webstore/search/graphiql%20feen

Postman
DeltaV SaaS SCADA recommends using Postman for testing and proof of concept work against the DeltaV SaaS SCADA GraphQL API.

https://www.getpostman.com/

Insomnia
Insomnia is a powerful web API tool that works well with GraphQL. You may prefer this tool to GraphIQL Feen for testing your queries.

https://insomnia.rest/

Language Specific GraphQL Client Libraries
You will also require a specific implementation of a GraphQL client in your language of choice.
Click here for more information on GraphQL

https://graphql.org/learn/

GraphQL Endpoints

Depending on where your accounts and credentials have been created for you, use one of the two endpoints identified:

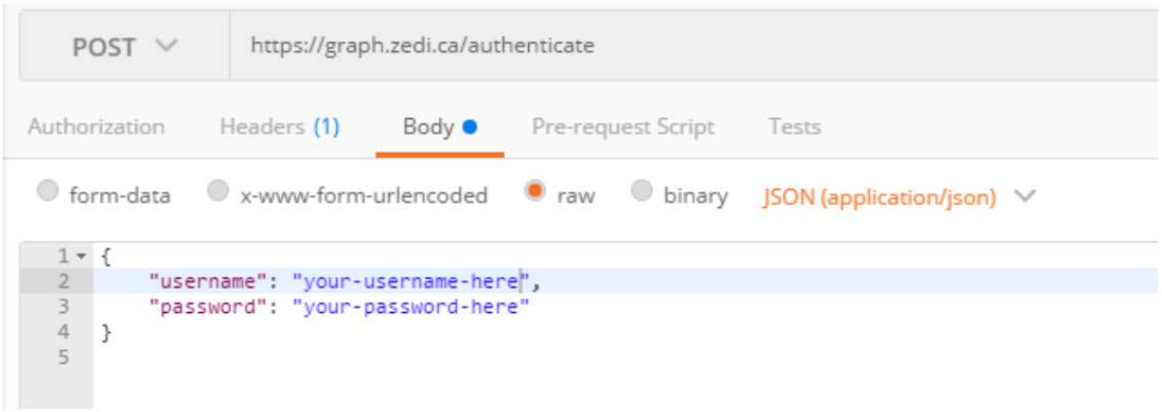
Environment	Endpoint
Test	https://graph.zediaccess.net/
Production	https://graph.zedi.ca/

Usage

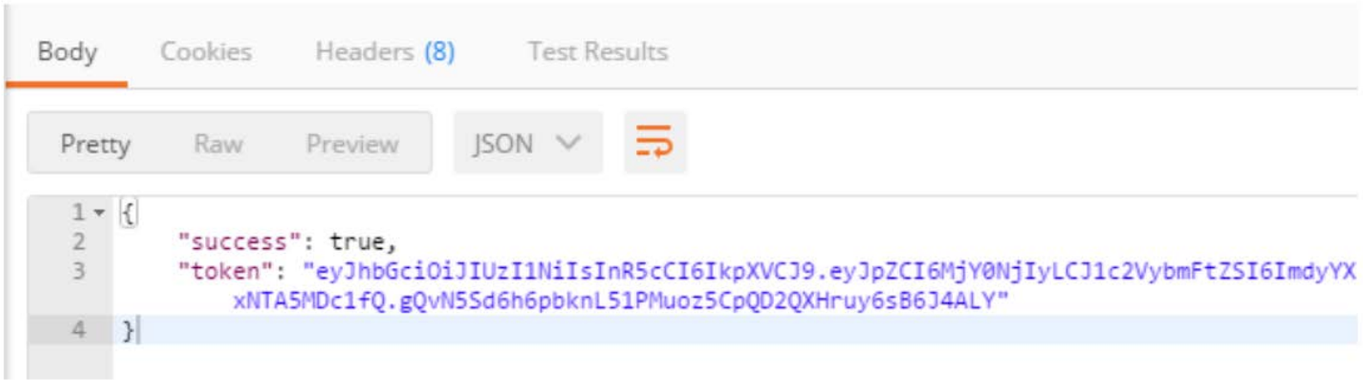
Authentication
Using postman, make a POST call to your-endpoint/authenticate:

```
Headers
POST /authenticate HTTP/1.1
Content-Type: application/json
Cache-Control: no-cache

Body
{
  "username": "your-username",
  "password": "your-password"
}
```



The response JSON will return a success indicator, and if successful a JWT that you will supply in the header of all future query calls.



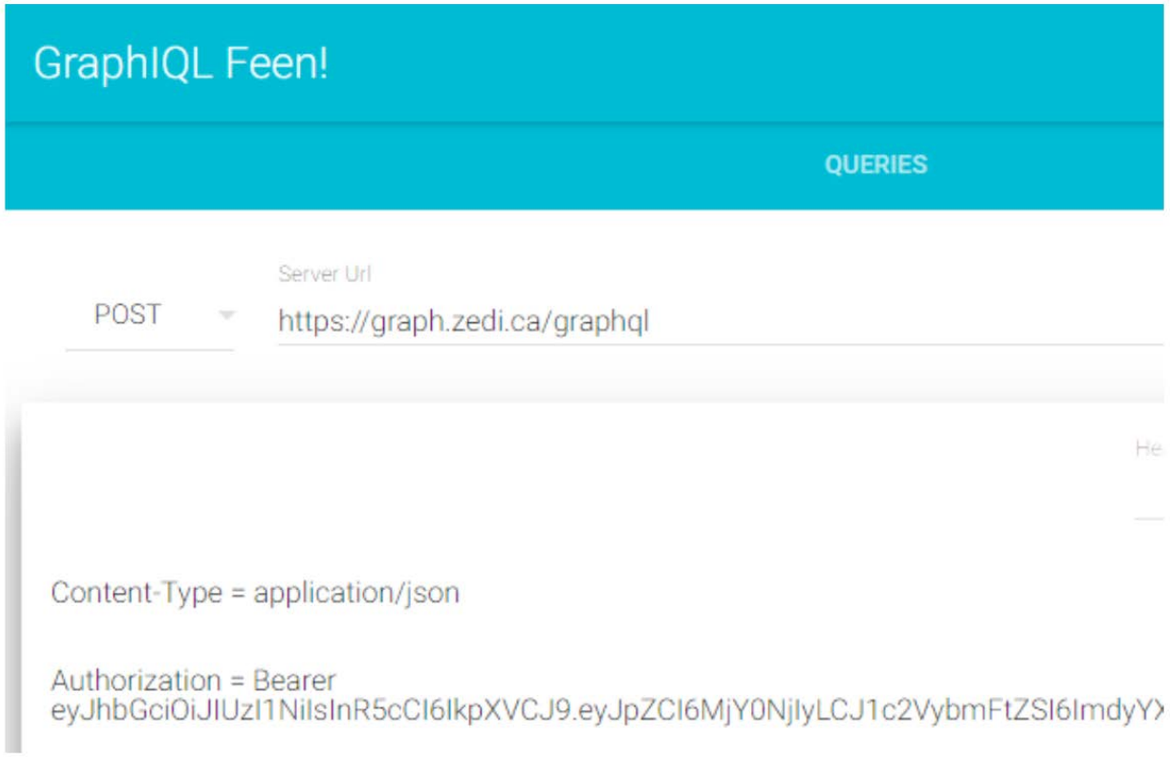
Configuring GraphIQL FEEN

GraphQL is a self documenting API through introspection. There is no external API reference required. Using a tool such as GraphIQL Feen, you may explore the current API implementation directly.

Open up GraphIQL Feen within Chrome, and click the “Servers” tab. Change the method to POST, and set the URL to your-endpoint/graphql

Add the following headers to the call:

```
Content-Type: application/json
Authorization: Bearer your-jwt-token-here
```

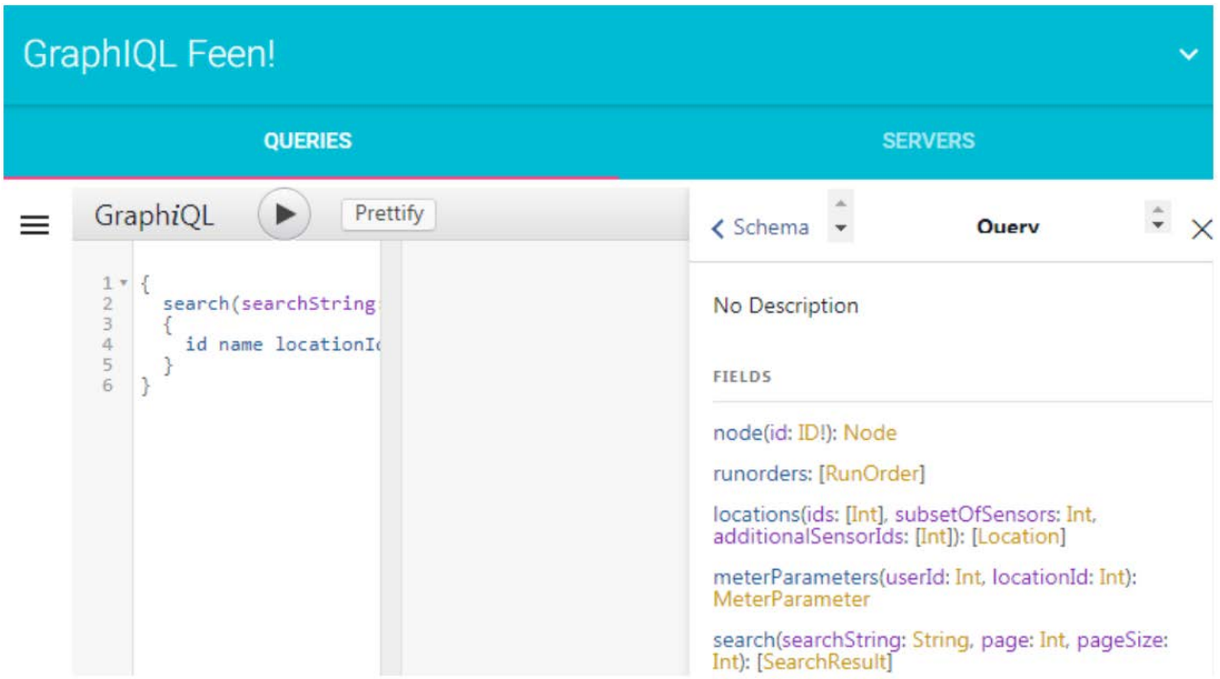


Create an Assets Query

On GraphIQL Feen, open up the “DOCS” pull out menu on the right side of the screen to see the introspection information. (if this is empty / blank spinner after initial setup, then just F5 the entire page to have it try introspection again).

Create an Assets Query (Continued)

Drill into the Query Root Type:



Observe the search Type:

```
search(searchString: Stringpage: IntpageSize: Int): [SearchResult]
```

You can drill down into the SearchResult object to see what properties are available. To get the name and ID for all locations you have access to, you can write a query like:

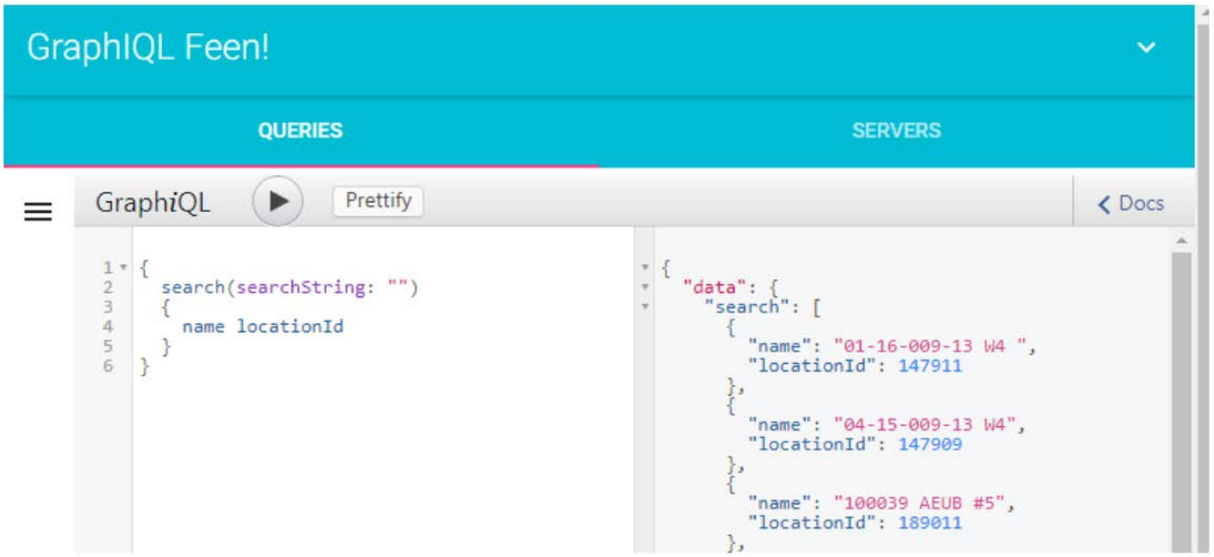
```
{
 search(searchString: "")
 {
  name locationId
 }
}
```

And you will receive a response such as:

```
{
 "data": {
  "search": [
   {
    "name": "Some Asset",
    "locationId": 12345
   },
   {
    "name": "Some Other Asset",
    "locationId": 12346
   }
  ]
 }
}
```

Create an Assets Query (Continued)

This is an example of a query you would pass to your language 's GraphQL client:



Create a Details Query

You can then get details on sensors for one (or more) of these locations using locations (ids: [Int]subsetOfSensors: Int): [Location]

```
{
 locations(ids: [12345] )
 {
  locationId name lastTransmissionDate
  coreSensors {name sensorId lastDataValueWithUnitOfMeasure lastDataDateTimeLocal}
webEnabledSensorsExcludingCore {name sensorId lastDataValueWithUnitOfMeasure
lastDataDateTimeLocal}
 }
}
```

Continued...

Create a Details Query (Continued)

This query will return data similar to:

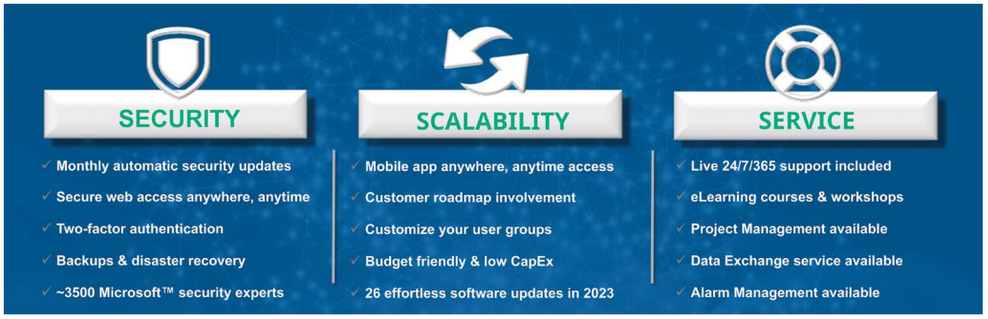

```
{
 "data": {
  "locations": [
   {
    "locationId": 12345,
    "name": "Some Asset",
    "lastTransmissionDate": "2018-06-20T19:21:09",
    "coreSensors": [
     {
      "name": "Gas Flow Rate",
      "sensorId": 1955550,
      "lastDataValueWithUnitOfMeasure": "16143.86 m3/Day",
      "lastDataDateTimeLocal": "2018-06-20 13:20:56 (MDT)"
     },
     {
      "name": "Static Pressure",
      "sensorId": 1955551,
      "lastDataValueWithUnitOfMeasure": "2406.82 kPa",
      "lastDataDateTimeLocal": "2018-06-20 13:20:56 (MDT)"
     }
  ]
 }
}
```


Create a History Query

You could also ask for historical data on one of these sensors as well:

```
{
 getSensorReadingReadings(
        fromDate:"2017-06-20T10:00:00Z",
        toDate:"2017-06-21T11:00:00Z",
                sensorIds: [1955550]) {readings{Amount UomChar ReadingTime}}
}
```

DeltaV™ SaaS SCADA is an IIoT cloud-native platform designed to enable asset-intensive industries to quickly and securely connect, acquire analytics and provide control of remotely located devices anywhere, anytime by anyone of your authorized users. We help our customers become more productive, profitable and sustainable to improve life around the globe.



DeltaVSaaS.Support@Emerson.com | 866 732 6967 | Global Support

North America and Latin America Global Headquarters
Emerson Automation Solutions
6005 Rogerdale Road
Houston, TX, USA 77072
T: +1 281 879 2699

Europe
Emerson Automation Solutions
Unit 1, Waterfront Business Park
Dudley Road, Drierley Hill
Dudley, UK DY5 1LX
T: +44 1384 487200

Middle East and Africa
Emerson Automation Solutions
PO Box 17033
Jebel Ali Free Zone - South 2
Dubai, UAE
T: +971 4 8118100

Asia Pacific
Emerson Automation Solutions
1 Panda Crescent
Singapore 128461
T: +65 6777 8211

Find us in social media

Emerson | DeltaV Automation Platform

DeltaV SaaS SCADA

DELTAV™

EMERSON™