

# DataGlen Platform Overview

## Table of Contents:

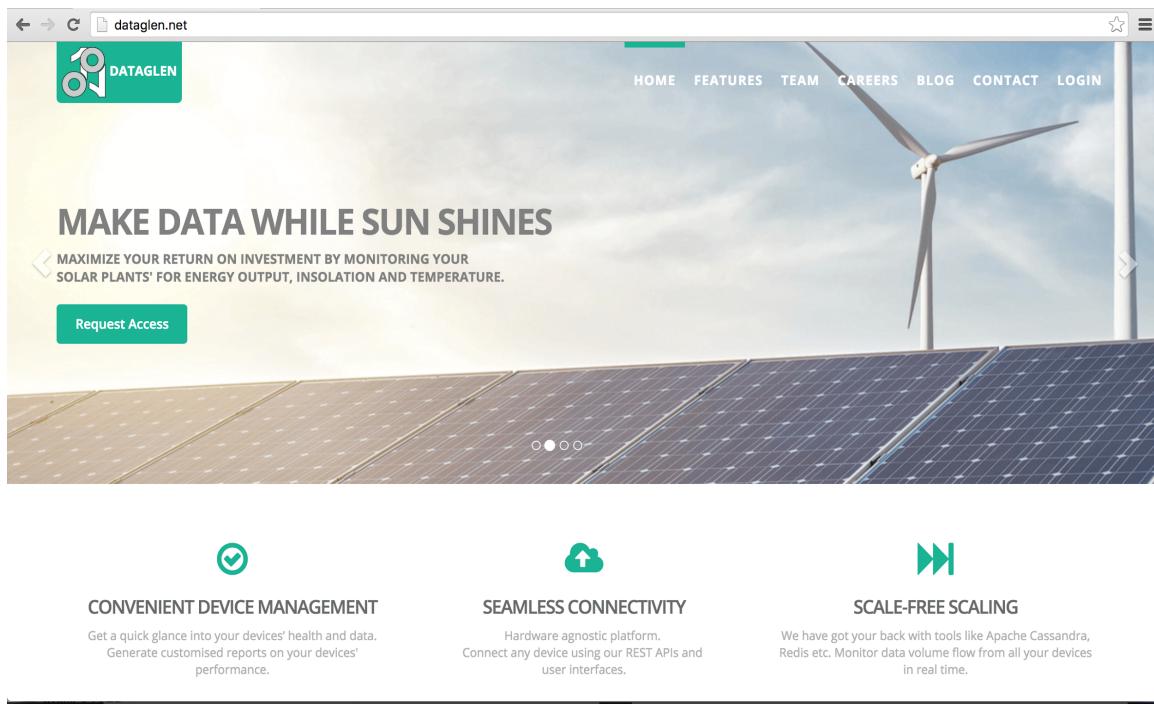
1	DataGlen Platform Uesr Interface	1
1.1	Login	1
1.2	Dashboard	3
1.3	Profile Page	4
1.4	Pending Devices	5
1.5	Create New Source	6
1.6	Sources Page:	7
1.7	Source Details	8
1.8	Defining New Data Stream	9
1.9	Define New Action Stream	10
1.10	Define New Configuration Stream	11
1.11	Nebula Dashboard	12
2	DataGlen REST APIs:	13
2.1	API's related to sources:	14
2.2	API's related to Data Streams:	25
2.3	API's related to Data Management	35
2.4	API's related to Action Management	42

# 1 DataGlen Platform Uesr Interface

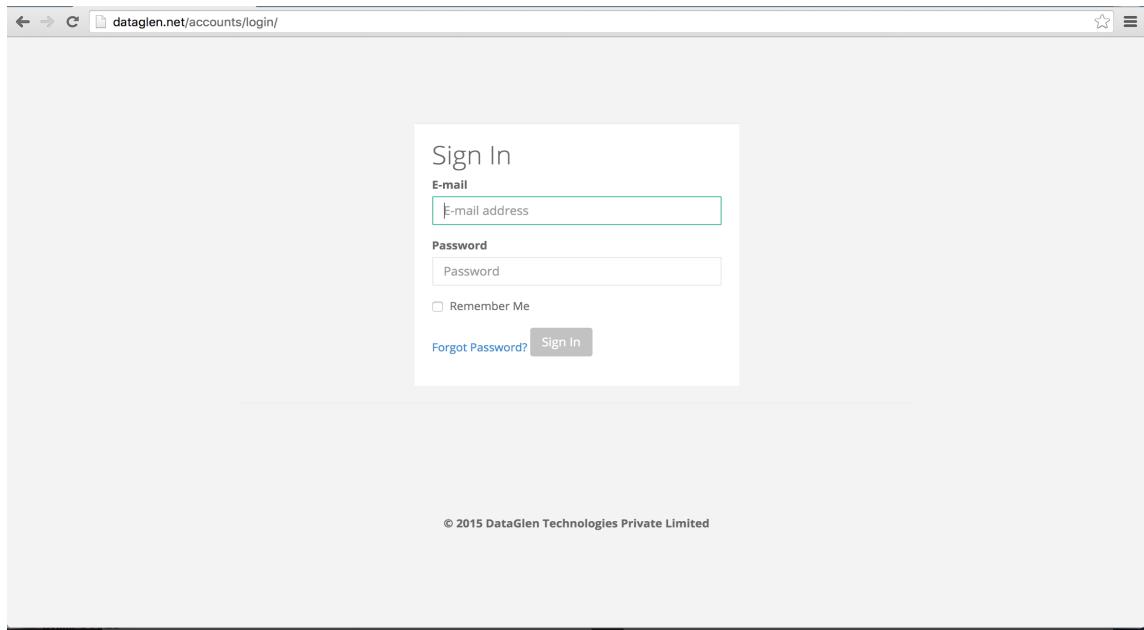
## 1.1 Login

Login to DataGlen IoELab platform: <http://dataglen.net/>

You will be directed to the DataGlen homepage which looks like below image:



Once you click on the **LOGIN** button on top right corner and you will be redirected to the link: <http://dataglen.net/accounts/login/> and the page will look like below image



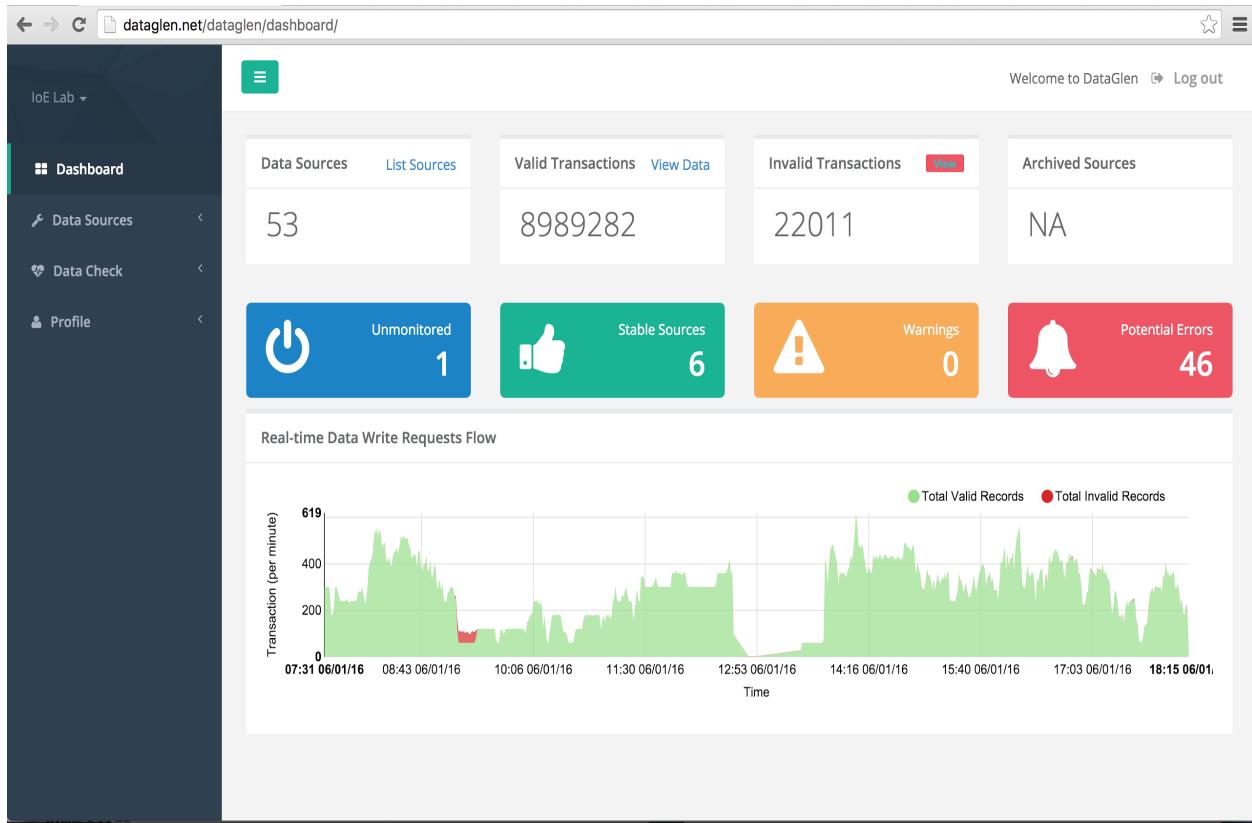
---

You will then enter your login credentials and click on “**Sign in**” button.  
If the credentials entered are correct, you’ll be directed to Dashboard page.

## 1.2 Dashboard

After entering the correct credentials on the login page, you will be directed to the the link <http://dataglen.net/dataglen/dashboard/>

i.e. the Dashboard, and the page looks like the below image:



This page contains the information about the sources.

It will also display information about transactions done on the sources.

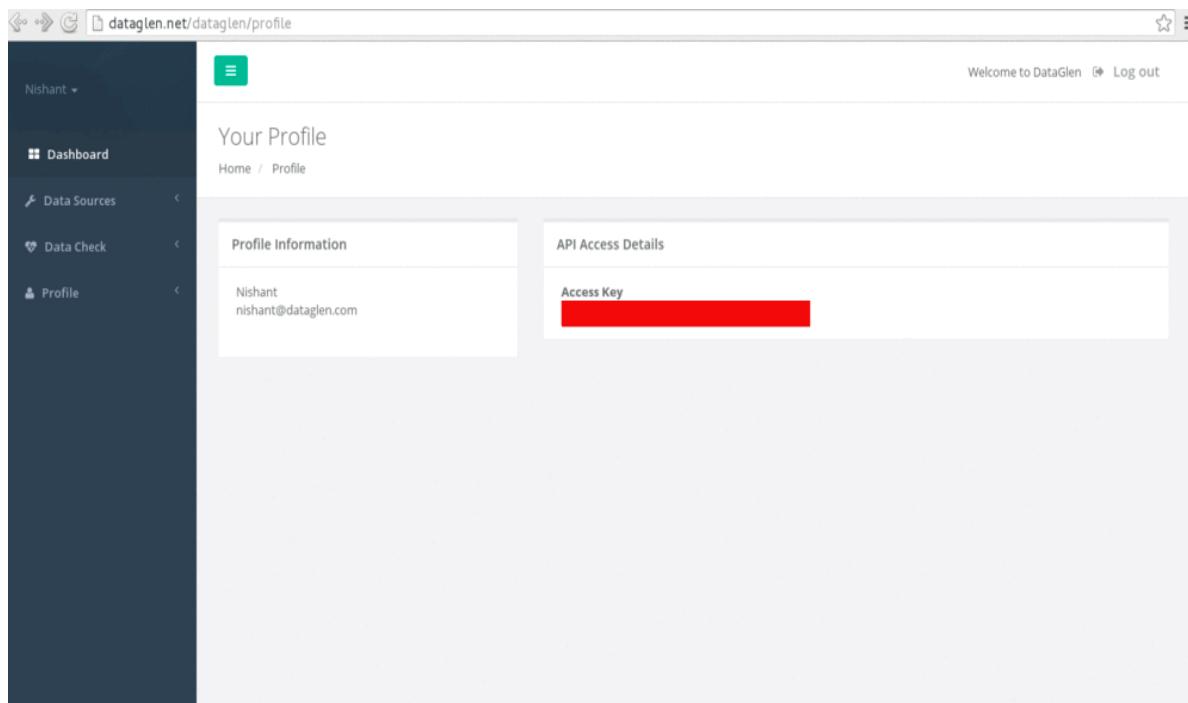
It will show all valid and invalid transactions done on the sources, that is nothing but the valid and invalid data write requests.

It will also show you a graph which shows the current real time data flowing-in.

## 1.3 Profile Page

From Dashboard page, you can go to your profile page by clicking on “**Profile**” from the left side menu. From here, you can get your API key that is required to be provided for integrating with DataGlen APIs .

This page will look like the below image

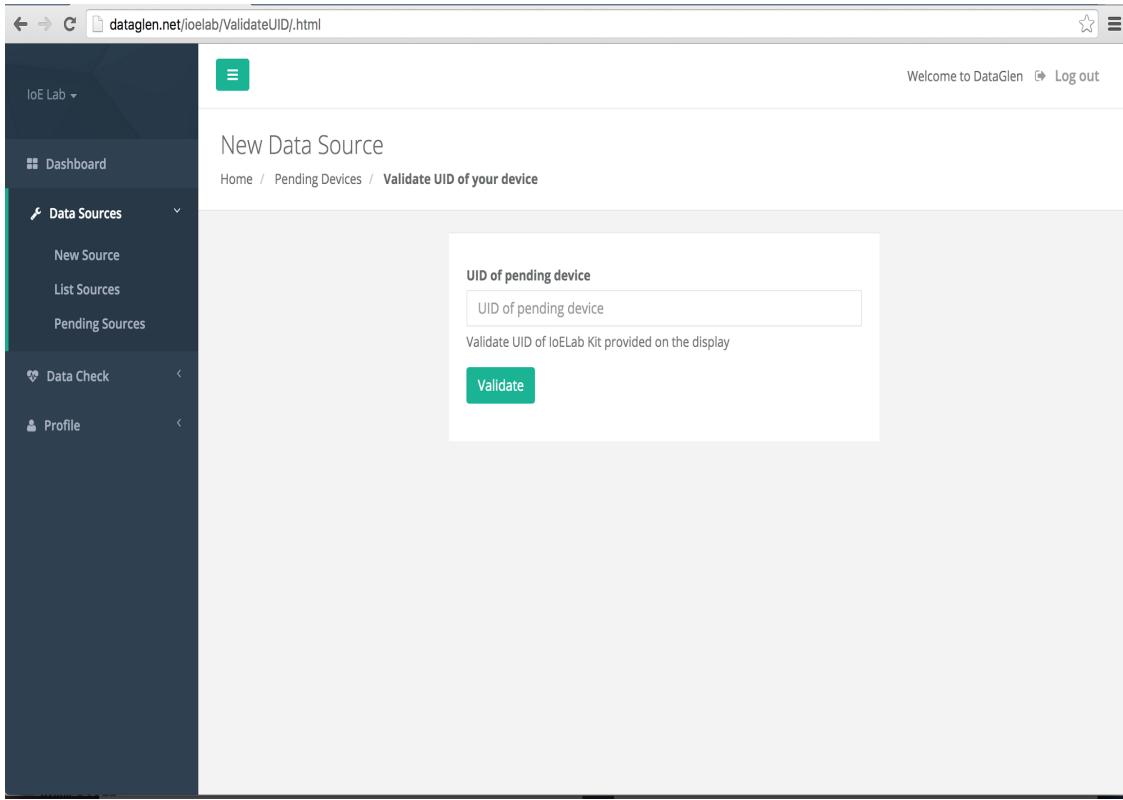


## 1.4 Pending Devices

If you would like to validate UID of your device, then you can chose the “Pending Sources” option under “Data Sources” from the left side menu.

You’ll be directed to the link:

<http://dataglen.net/ioelab/ValidateUID/.html> and the web page looks like the below image:



Enter the UID of your device and click on the Validate button to authenticate the source

## 1.5 Create New Source

A New Source can be added by selecting “Add New Source” option under Data Sources from the left side of the menu.

You will be directed to the link: <http://dataglen.net/dataglen/create/source/>

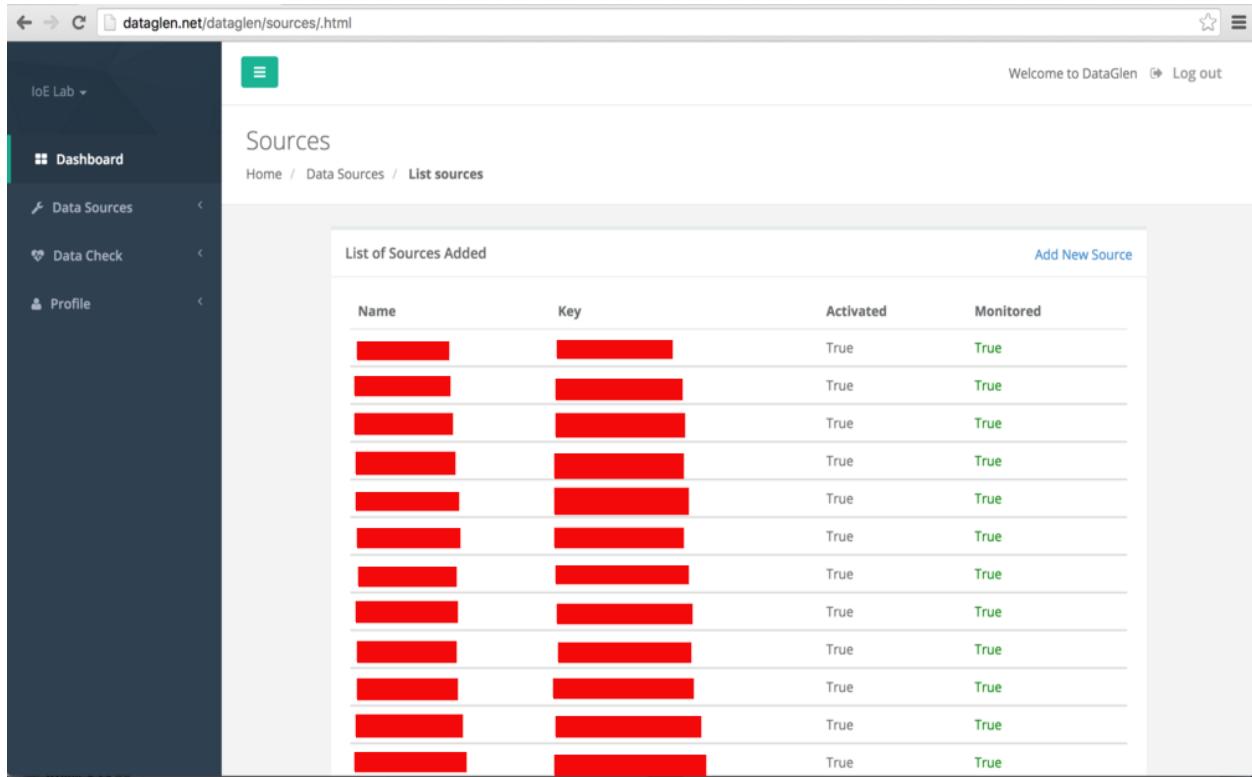
And the page looks like the below image:

The screenshot shows a web browser window with the URL <http://dataglen.net/dataglen/create/source/> in the address bar. The page title is "New Data Source". The left sidebar has a dark theme with options: IoE Lab (dropdown), Dashboard, Data Sources (selected), Data Check, and Profile. The main content area shows the "Add a new Source" form. It includes fields for Name (placeholder: "Name of the data source"), Data Reporting Interval (placeholder: "In Seconds"), Source Identifier (e.g. MAC) (placeholder: "MAC address of the source. Optional."), Data Format (dropdown menu showing "-----"), UID (placeholder: "UID"), and Response Text with HTTP 200 OK. Navigation links at the top include Home, Data Sources, Add a new Source, and a breadcrumb trail: Home / Data Sources / Add a new Source. The top right corner shows "Welcome to DataGlen" and "Log out".

## 1.6 Sources Page:

You can view all the sources, by clicking on the “List Sources” option on the Dashboard page, or by selecting “List Sources” option under “Data Sources” from the left side menu.

After selecting the List Sources option, you will be directed to the link:  
<http://dataglen.net/dataglen/sources/>; the page looks like below image:



The screenshot shows a web browser window with the URL <http://dataglen.net/dataglen/sources/.html>. The page has a dark blue sidebar on the left with options: IoE Lab, Dashboard, Data Sources (selected), Data Check, and Profile. The main content area has a header "Sources" and a breadcrumb "Home / Data Sources / List sources". Below this is a table titled "List of Sources Added" with columns: Name, Key, Activated, and Monitored. There are 10 rows in the table, each with red bars in the Name and Key columns. The Activated and Monitored columns show "True" in green. At the top right of the table is a button "Add New Source".

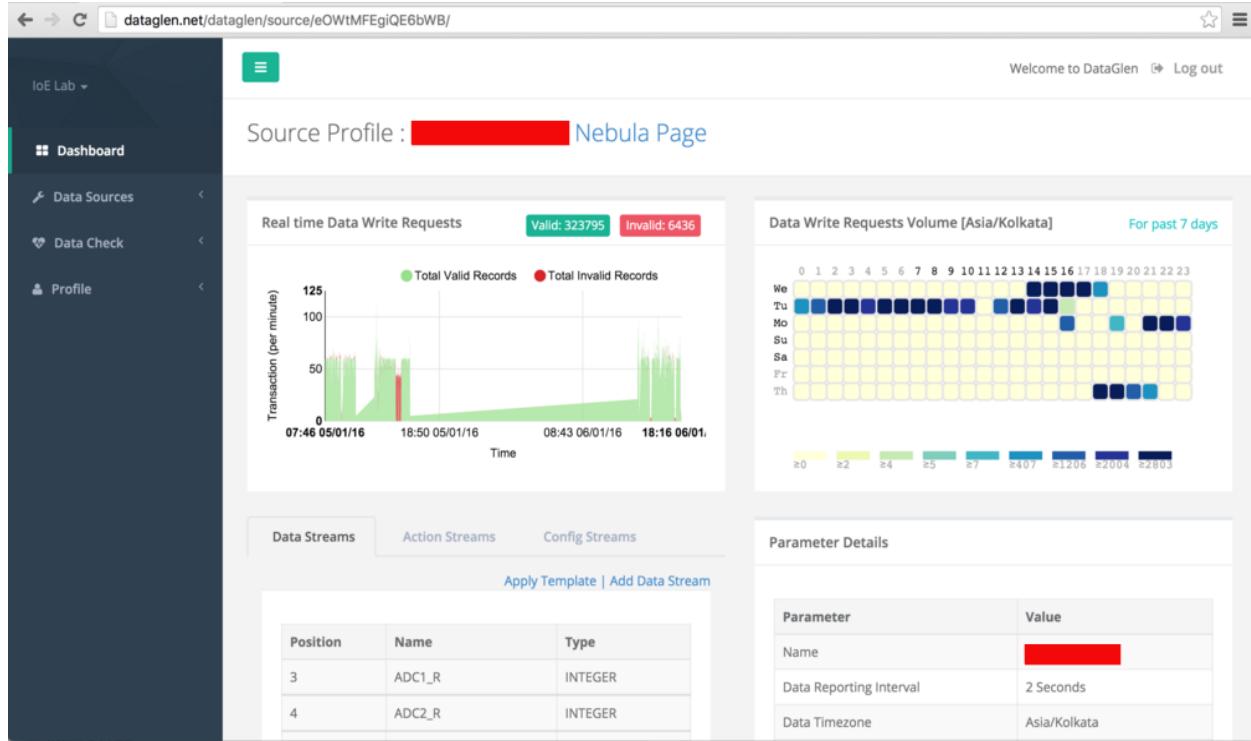
List of Sources Added				Add New Source
Name	Key	Activated	Monitored	
[Red Bar]	[Red Bar]	True	True	
[Red Bar]	[Red Bar]	True	True	
[Red Bar]	[Red Bar]	True	True	
[Red Bar]	[Red Bar]	True	True	
[Red Bar]	[Red Bar]	True	True	
[Red Bar]	[Red Bar]	True	True	
[Red Bar]	[Red Bar]	True	True	
[Red Bar]	[Red Bar]	True	True	
[Red Bar]	[Red Bar]	True	True	

New Source can also be added by clicking on Add New Source option on sources page.

## 1.7 Source Details

You can view the detailed profile of all the sources, by then clicking either on the source name or the key from the sources page, you will be directed  
The url:<http://dataglen.net/dataglen/source/eOWtMFEgiQE6bWB/>  
Where “eOWtMFEgiQE6bWB” is the source key.

The web page looks like the below image:



The source profile page contains all the details about the source.  
It shows all the parameter details of the source and all the data, action and configuration streams of the source.  
The graphs on this page show the real data flowing-in and also the data write requests volume for the past 7 day.

## 1.8 Defining New Data Stream

New data streams can be added for the source by clicking the “Add Data Stream” link under Data Streams tab. You will be directed to the link:

<http://dataglen.net/dataglenstreams/create/eOWtMFEgiQE6bWB/>

and the page looks like the below image:

The screenshot shows a web browser window with the URL <http://dataglen.net/dataglenstreams/create/eOWtMFEgiQE6bWB/> in the address bar. The page title is "New Data Stream". The left sidebar has a dark theme with options: IoE Lab, Dashboard (selected), Data Sources, Data Check, and Profile. The main content area is titled "New Data Stream" and includes the following fields:

- Name:** A text input field labeled "Name of the DataStream" with the placeholder "Name of the data stream, should be unique for this source".
- StreamDataType:** A dropdown menu currently showing "-----". A tooltip below it says "Type of the data this stream reports".
- Date/Time fields format:** A text input field labeled "Date/Time fields format." with the placeholder "DateTime field values should be reported in ISO 8601 formats. You can specify custom formats here."
- StreamPositionInCSV:** A text input field labeled "Position of the stream" with the placeholder "Applicable only if source dataFormat is CSV, should be unique".
- StreamDataUnit:** A text input field labeled "Data unit (Optional)" with the placeholder "Data Unit".

## 1.9 Define New Action Stream

Similar to add streams, the action streams can be added by clicking, “Add Action Stream” link under “Action Streams” tab from the source profile page. After doing so, you will be directed to the link:

[http://dataglen.net/dataglen/action\\_streams/create/eOWtMFEgiQE6bWB/](http://dataglen.net/dataglen/action_streams/create/eOWtMFEgiQE6bWB/)

And the page looks like the below image:

The screenshot shows a web browser window with the URL [http://dataglen.net/dataglen/action\\_streams/create/eOWtMFEgiQE6bWB/](http://dataglen.net/dataglen/action_streams/create/eOWtMFEgiQE6bWB/). The page is titled "New Action Stream". On the left, there is a sidebar with "IoE Lab" at the top, followed by "Dashboard", "Data Sources", "Data Check", and "Profile". The main content area has a header "New Action Stream" and a breadcrumb "Home / Action Streams / Add a new Action Stream". Below this, there are several input fields and descriptions:

- Name:** A text input field containing "Name of the Action DataStream". A placeholder text "Name of the data stream, should be unique for this source" is visible below the input.
- StreamDataType:** A dropdown menu currently showing "-----". A placeholder text "Type of the data this stream reports" is visible below the dropdown.
- Date/Time fields format:** A text input field containing "Date/Time fields format.". A placeholder text "DateTime field values should be reported in ISO 8601 formats. You can specify custom formats here." is visible below the input.
- StreamPositionInCSV:** A text input field containing "Position of the stream". A placeholder text "Applicable only if source dataFormat is CSV, should be unique" is visible below the input.
- StreamDataUnit:** A text input field containing "Data unit (Optional)". A placeholder text "Data Unit" is visible below the input.

## 1.10 Define New Configuration Stream

Similarly, action streams, “Add Config Stream” link can be clicked under “Config Streams” tab from the source profile page. After doing so, You will be directed to the link:

[http://dataglen.net/dataglen/config\\_streams/create/eOWtMFEgiQE6bWB/](http://dataglen.net/dataglen/config_streams/create/eOWtMFEgiQE6bWB/)

and the page looks like the below image:

The screenshot shows a web browser window with the URL [http://dataglen.net/dataglen/config\\_streams/create/eOWtMFEgiQE6bWB/](http://dataglen.net/dataglen/config_streams/create/eOWtMFEgiQE6bWB/). The page title is "New Config Stream". The left sidebar has sections for IoE Lab, Dashboard, Data Sources, Data Check, and Profile. The main content area shows fields for creating a new config stream:

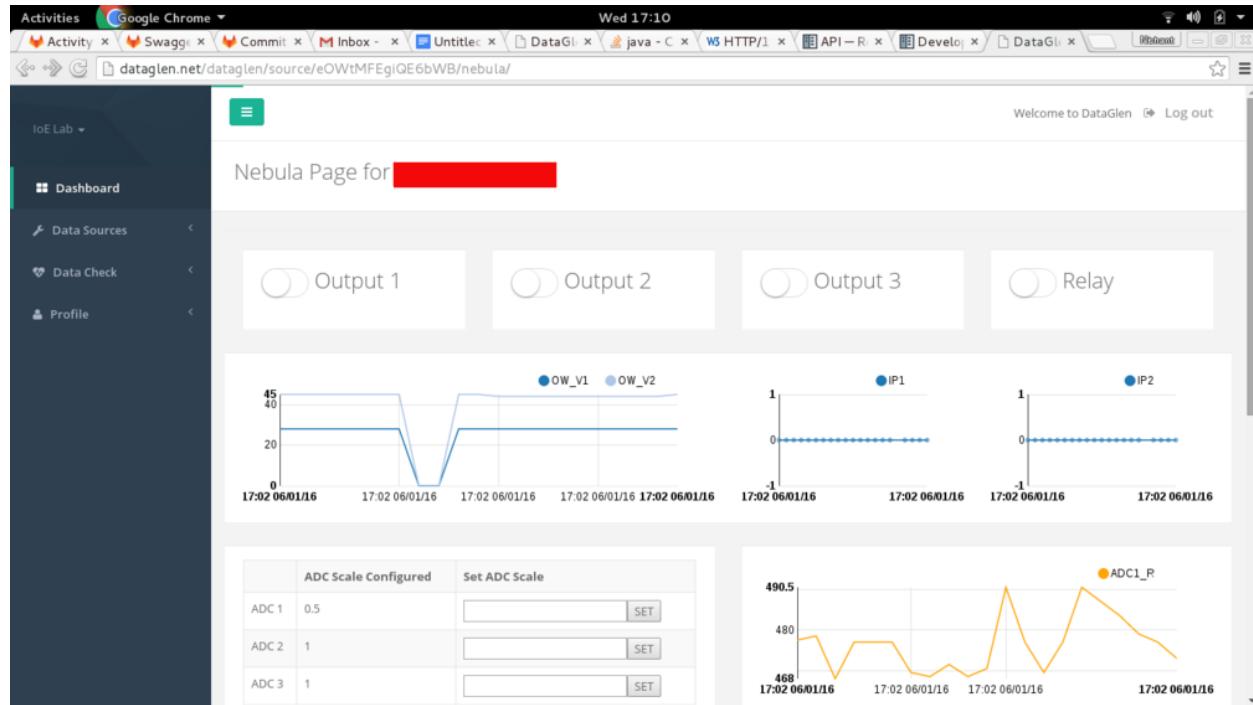
- Name:** Name of the Action DataStream  
Name of the data stream, should be unique for this source
- StreamDataType:**   
Type of the data this stream reports
- Date/Time fields format.**   
DateTime field values should be reported in ISO 8601 formats. You can specify custom formats here.
- StreamPositionInCSV:**   
Applicable only if source dataFormat is CSV, should be unique
- StreamDataUnit:**   
Data Unit

## 1.11 Nebula Dashboard

You can visit the Nebula dashboard by clicking the “Nebula Page” link on the source details page. After doing so, you will be directed to the link:

<http://dataglen.net/dataglen/source/eOWtMFEgiQE6bWB/nebula/>

And the page looks like the below images:





This page contains graphs of various parameters of the nebula board. Only from this page, actuation control can be gained over the device.

## 2 DataGlen REST APIs:

The DataGlen Platform REST API's can be accessed from the link:

<http://dataglen.net/api/docs/>

This page contains all the API's related to sources, streams, actions and configs under different sections.

This page looks like the below image:

The screenshot shows a web browser window with the URL [dataglen.net/api/docs/#!/Solar\\_Plants/Plants\\_list](http://dataglen.net/api/docs/#!/Solar_Plants/Plants_list). The page title is "REST APIs Documentation". On the left, there's a sidebar with a logo and sections for "TOKEN" (with a text input field and a button), "API REFERENCE" (listing "Data Sources", "Data Streams", "Data Management", "Action Streams", and "Action Management"), and a "TRY" button for API calls. The main content area has a heading "Returns a list of sources that are owned by you" with a "GET /api/sources/" link. It includes a "Show samples" button, a "Manage your data sources on the DataGlen platform." note, and a "Returns a list of sources that are owned by you." note. Below this is a "Test this endpoint" section with a "TRY" button and a "Response Type" dropdown set to "application/json". A large dark gray box covers the right side of the main content area.

You need to enter your API key and after that all the API's can be accessed. Request and Response body for all the API's can be found by clicking show samples icon for each API.

## 2.1 API's related to sources:

### 2.1.1 Get a list of sources owned by you:

#### **API: /api/sources (GET)**

You can view all the sources owned by you using above API

**Parameters:** None

**Response:**

**name (string)**: Name of the source, it should be unique,  
**sourceKey (string)**: Unique sensor key, a read only field,  
**UID (string)**: If UID is specified, it should be unique across the entire platform.,  
**dataFormat (choice) = ['JSON' or 'CSV']**: The format in which this source sends data records,  
**dataReportingInterval (integer)**: Data reporting interval (in seconds),  
**sourceMacAddress (string)**: Source hardware identifier,  
**textMessageWithHTTP200 (string)**: Text message to be accompanied with HTTP 200 OK,  
**textMessageWithError (string)**: Text message to be accompanied with an error,  
**isActive (boolean)**: Data will be accepted only if this parameter is True,  
**isMonitored (boolean)**: This source will be monitored only if this parameter is True,  
**actuationEnabled (boolean)**: If this source has actuation features enabled. Keep it unchecked if unsure.,  
**csvDataKeyName (string)**: Key name that holds comma separated data (applicable only if source dataFormat is CSV),  
**timeoutInterval (integer)**: If there is no data for these many seconds, a notification will be raised,  
**dataTimezone (choice)** = Choice of different time zones

### Error Status Codes

HTTP Status Code	Reason
200	Success
401	Not authenticated

This API looks like the below image on the API docs page:

**TOKEN**

Returns a list of sources that are owned by you

API REFERENCE

Data Sources

Creates a new data source  
Delete a data source  
Get a source with the mentioned ...  
Update an existing data source

Data Streams

Data Management

Action Streams

Action Management

Config Streams

Config Management

GET /api/sources/

Manage your data sources on the DataGlen platform.

Returns a list of sources that are owned by you.

Test this endpoint

TRY

Response Type application/json

RESPONSE SAMPLE

```
{
  "name": "string",
  "sourceKey": "string",
  "dataFormat": "JSON",
  "dataReportingInterval": 0,
  "UID": "string",
  "actuationEnabled": true,
  "csvDatakeyName": "string",
  "dataTimezone": "Africa/Abidjan",
  "isActive": true,
  "isMonitored": true,
  "sourceMacAddress": "string",
  "textMessageWithTheError": "string",
  "textMessageWithHTTP200": "string",
  "timeoutInterval": 0
}
```

RESPONSE SCHEMA (CLICK TO EXPAND)

This API can be accessed directly from the code in any language.  
Below is the code sample to get the sources owned by you in python:

```
API = 'your API key'
# Get list of data sources
def get_data_source(self):
    self.server_address = 'http://www.dataglen.net'
    self.sources_url = '/api/sources/'
    self.api_key = API
    self.token = 'Token ' + self.api_key
    self.url = self.server_address + self.sources_url
    self.auth_header = {'Authorization': self.token}
    response = requests.get(self.url, headers = self.auth_header)
    print(response.status_code)
```

## 2.1.2 Create a new Data Source

### API: /api/sources (POST)

**Description:** A new data source can be created using above API:

#### Parameters:

Parameter Name	Parameter Type	Data Type
body	body	body

#### Response:

**name (string):** Name of the source, it should be unique,  
**sourceKey (string):** Unique sensor key, a read only field,  
**UID (string):** If UID is specified, it should be unique across the entire platform.,  
**dataFormat (choice) = ['JSON' or 'CSV']:** The format in which this source sends data records,  
**dataReportingInterval (integer):** Data reporting interval (in seconds),  
**sourceMacAddress (string):** Source hardware identifier,  
**textMessageWithHTTP200 (string):** Text message to be accompanied with HTTP 200 OK,  
**textMessageWithError (string):** Text message to be accompanied with an error,  
**isActive (boolean):** Data will be accepted only if this parameter is True,  
**isMonitored (boolean):** This source will be monitored only if this parameter is True,  
**actuationEnabled (boolean):** If this source has actuation features enabled. Keep it unchecked if unsure.,  
**csvDataKeyName (string):** Key name that holds comma separated data (applicable only if source dataFormat is CSV),  
**timeoutInterval (integer):** If there is no data for these many seconds, a notification will be raised,  
**dataTimezone (choice) =** Choice of different time zones

### Error Status Codes

HTTP Status Code	Reason
---------------------	--------

201	Success. Accompanied with details of the new source.
400	If any of the essential parameters are missing (BAD_REQUEST).
409	If a source with the mentioned name already exists (DUPLICATE_SOURCE).
401	Not authenticated

This API looks like the below image on the API docs page:

The screenshot shows the DataGlen API documentation for the `/api/sources/` endpoint. The left sidebar contains a 'TOKEN' input field and a 'API REFERENCE' section with links to various DataGlen services. The main content area has a title 'Create a new data source' and a 'POST /api/sources/' button. Below this is a description: 'Manage your data sources on the DataGlen platform.' and 'Create a new data source.' A 'Parameters' section shows a 'body' input field with the note '(required)' and a 'Content type:' dropdown set to 'application/json'. Below the input field is a note: 'Details of the new source to be created.' To the right, there is a 'RESPONSE SAMPLE' section displaying a JSON schema for the response:

```
{
  "name": "string",
  "sourcekey": "string",
  "dataFormat": "JSON",
  "dataReportingInterval": 0,
  "UID": "string",
  "actuationEnabled": true,
  "csvDataKeyName": "string",
  "dataTimeZone": "Africa/Abidjan",
  "isActive": true,
  "isMonitored": true,
  "sourceMacAddress": "string",
  "textMessageWithErr": "string",
  "textMessageWithHTTP200": "string",
  "timeoutInterval": 0
}
```

Below the sample is a 'RESPONSE SCHEMA (CLICK TO EXPAND)' link and a 'BODY SAMPLE' section showing a simplified version of the JSON schema.

This API can be accessed directly from the code in any language.

Below is the code sample to get the sources owned by you in python:

```
API = 'your API key'
# Adding a data source
def create_data_source(self):
    self.server_address = 'http://www.dataglen.net'
    self.sources_url = '/api/sources/'
```

```

self.api_key = API
self.token = 'Token ' + self.api_key
self.url = self.server_address + self.sources_url
self.post_body = {
    "name": "test_source_csv",
    "dataFormat": "CSV",
    "dataReportingInterval": 10,
    "isActive": False,
    "isMonitored": True
}
length = sys.getsizeof(self.post_body)
self.auth_header = {'Authorization': self.token, 'content-length': length, 'content-type': 'application/json'}
response = requests.post(self.url, data = json.dumps(self.post_body), headers = self.auth_header)
print(response.status_code)

```

### 2.1.3 Delete a data source:

**API: /api/sources/{key} (DELETE)**

**Description:** A data source can be deleted using above API.

**Parameters:**

Parameter Name	Parameter Type	Data Type
key	path	string

### Error Status Codes

HTTP Status Code	Reason
204	Success. The source has been deleted.
400	If the specified source key does not exist (INVALID_SOURCE_KEY).
401	Not authenticated

This API looks like the below image on the API docs page:

The screenshot shows a web browser window displaying the DataGlen API documentation. The URL in the address bar is `dataglen.net/api/docs/#!/Data_Sources/Sources_destroy`. The main content area is titled "Delete a data source". It includes a "DELETE" button and the endpoint URL `/api/sources/{key}/`. Below this, there is a brief description: "Manage your data sources on the DataGlen platform." and a note: "Delete a data source." A "Parameters" section contains a field labeled "key" with the note "(required)" and "string". There is also a "TRY" button and a "Response Type" dropdown set to "application/json". To the right, there is a "RESPONSE SAMPLE" section showing an empty JSON object {} and a "RESPONSE SCHEMA (CLICK TO EXPAND)" section which is currently collapsed. On the left side, there is a sidebar titled "TOKEN" with a redacted token input field and a "Show samples" button. The sidebar also lists various API reference categories such as Data Sources, Data Streams, Data Management, Action Streams, Action Management, Config Streams, Config Management, and Solar Plants.

## 2.1.4 Get a source source with mentioned key:

**API: /api/sources/{key} (GET)**

**Description:** A data source can be deleted using above API.

**Parameters:**

Parameter Name	Parameter Type	Data Type
key	path	string

## **Response:**

**name (string):** Name of the source, it should be unique,  
**sourceKey (string):** Unique sensor key, a read only field,  
**UID (string):** If UID is specified, it should be unique across the entire platform.,  
**dataFormat (choice) = ['JSON' or 'CSV']:** The format in which this source sends data records,  
**dataReportingInterval (integer):** Data reporting interval (in seconds),  
**sourceMacAddress (string):** Source hardware identifier,  
**textMessageWithHTTP200 (string):** Text message to be accompanied with HTTP 200 OK,  
**textMessageWithError (string):** Text message to be accompanied with an error,  
**isActive (boolean):** Data will be accepted only if this parameter is True,  
**isMonitored (boolean):** This source will be monitored only if this parameter is True,  
**actuationEnabled (boolean):** If this source has actuation features enabled. Keep it unchecked if unsure.,  
**csvDataKeyName (string):** Key name that holds comma separated data (applicable only if source dataFormat is CSV),  
**timeoutInterval (integer):** If there is no data for these many seconds, a notification will be raised,  
**dataTimezone (choice) =** Choice of different time zones

## **Error Status Codes**

HTTP Status Code	Reason
200	Success
400	If the specified source key does not exist (INVALID_SOURCE_KEY).
401	Not authenticated

This API looks like the below image on the API docs page:

## 2.1.5 Update an existing data source:

**API: /api/sources/{key} (PATCH)**

**Description:** Details of any source can be updated using above API.

### Parameters:

Parameter Name	Parameter Type	Data Type
key	path	string
body	body	body

## **Response:**

**name (string):** Name of the source, it should be unique,  
**sourceKey (string):** Unique sensor key, a read only field,  
**UID (string):** If UID is specified, it should be unique across the entire platform.,  
**dataFormat (choice) = ['JSON' or 'CSV']:** The format in which this source sends data records,  
**dataReportingInterval (integer):** Data reporting interval (in seconds),  
**sourceMacAddress (string):** Source hardware identifier,  
**textMessageWithHTTP200 (string):** Text message to be accompanied with HTTP 200 OK,  
**textMessageWithError (string):** Text message to be accompanied with an error,  
**isActive (boolean):** Data will be accepted only if this parameter is True,  
**isMonitored (boolean):** This source will be monitored only if this parameter is True,  
**actuationEnabled (boolean):** If this source has actuation features enabled. Keep it unchecked if unsure.,  
**csvDataKeyName (string):** Key name that holds comma separated data (applicable only if source dataFormat is CSV),  
**timeoutInterval (integer):** If there is no data for these many seconds, a notification will be raised,  
**dataTimezone (choice)** = Choice of different time zones

## **Error Status Codes**

HTTP Status Code	Reason
201	Success. Accompanied with updated details of the source.
400	If any of the essential parameters are missing or invalid (BAD_REQUEST) or If the specified source key does not exist (INVALID_SOURCE_KEY).
409	If a source with the mentioned name already exists (DUPLICATE_SOURCE).
401	Not authenticated

This API looks like the below image on the API docs page:

The screenshot shows the DataGlen API documentation for the `PATCH /api/sources/{key}` endpoint. The left sidebar shows the API reference for Data Sources, including options to Create a new data source, Delete a data source, and Get a source with the mentioned key. The main content area is titled "Update an existing data source". It includes a "Parameters" section with a "key" field (required, string) and a "body" field (Content type: application/json, WriteSourceSerializer). The "RESPONSE SAMPLE" shows a JSON object with fields like name, sourceKey, dataFormat, dataReportingInterval, UID, actuationEnabled, csvDataKeyName, dataTimezone, isActive, isMonitored, sourceMacAddress, textMessageWithErr, textMessageWithHTTP200, and timeoutInterval. The "RESPONSE SCHEMA (CLICK TO EXPAND)" and "BODY SAMPLE" sections also show similar JSON structures.

This API can be accessed directly from the code in any language.

Below is the code sample to get the sources owned by you in python:

```
API='your API key'
SOURCE_KEY='source key of data source'
# Activate a data source/ update data source
def activate_data_source(self):
    self.server_address = 'http://www.dataglen.net'
    self.sources_url = '/api/sources/'
    self.api_key = API
    self.token = 'Token ' + self.api_key
    self.url = self.server_address + self.sources_url + SOURCE_KEY + '/'
    self.post_body = {
        "name": "test_source_csv",
        "dataFormat": "CSV",
        "dataReportingInterval": 10,
```

```

        "isActive": True,
        "isMonitored": True
    }
length = sys.getsizeof(self.post_body)
self.auth_header = {'Authorization': self.token, 'content-length': length, 'content-type': 'application/json'}
response = requests.patch(self.url, data = json.dumps(self.post_body), headers = self.auth_header)
print(response.status_code)

```

## 2.2 API's related to Data Streams:

### 2.2.1 Get a list data streams for a data source

**API: /api/sources/{source\_key}/streams (GET)**

**Description:** You can view all the sources owned by you using above API

**Parameters:**

Parameter Name	Parameter Type	Data Type
source_key	path	string

**Response:**

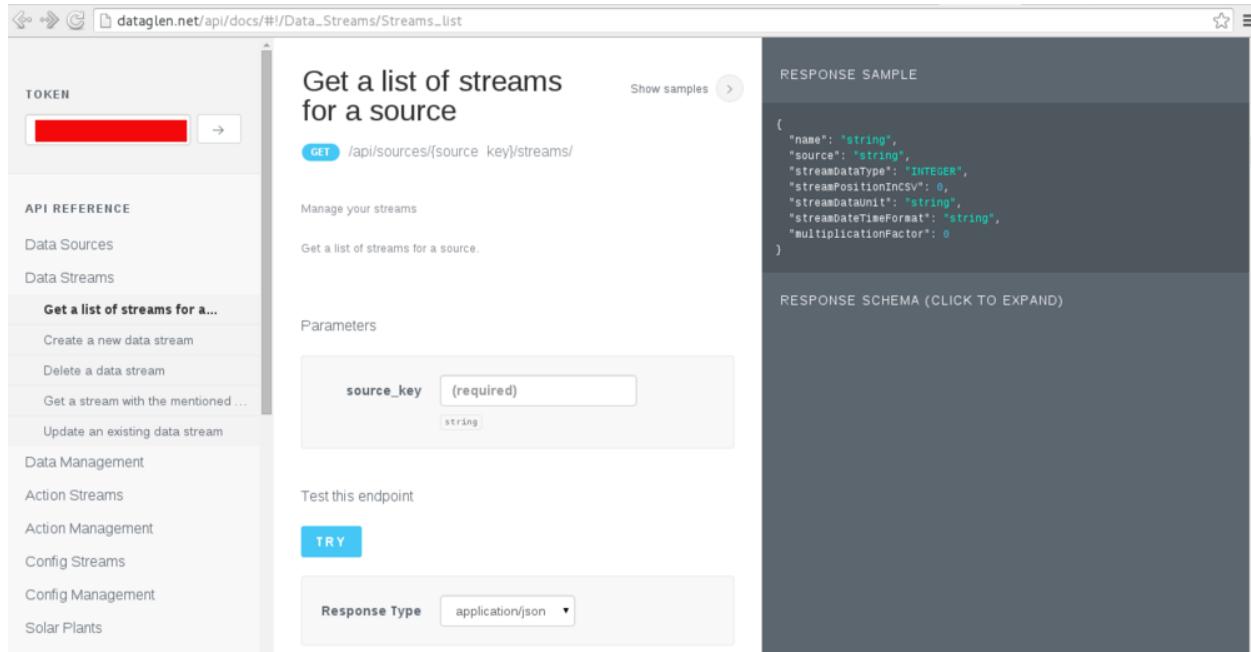
**name (string):** Name of the data stream, should be unique for this source,  
**source (string):** Key of the source this stream belongs to,  
**streamDataType (choice) = ['INTEGER' or 'BOOLEAN' or 'STRING' or 'FLOAT' or 'LONG' or 'MAC' or 'TIMESTAMP' or 'DATE' or 'TIME']:** Type of the data this stream reports,

**streamDataUnit (string):** Data Unit,  
**streamDateTimeFormat (string):** DateTime field values should be reported in ISO 8601 formats. You can specify custom formats here,  
**streamPositionInCSV (integer):** Applicable only if source dataFormat is CSV, should be unique,  
**multiplicationFactor (number):** Multiplication factor for this stream, applicable only for numerical streams. Defaults to one.

## Error Status Codes

HTTP Status Code	Reason
200	Success
400	If the source key mentioned is invalid (INVALID_SOURCE_KEY).
401	Not authenticated

This API looks like the below image on the API docs page:



The screenshot shows a web-based API documentation interface for the `/api/sources/{source_key}/streams/` endpoint. On the left, there's a sidebar with navigation links for TOKEN, API REFERENCE (Data Sources, Data Streams, etc.), and other management sections. The main content area has a title "Get a list of streams for a source" with a "Show samples" button. Below it is a "Parameters" section with a "source\_key" input field labeled "(required)" and type "string". There's also a "Test this endpoint" section with a "TRY" button and a "Response Type" dropdown set to "application/json". To the right, there's a "RESPONSE SAMPLE" showing a JSON object with fields like name, source, streamdatatype, streampositionincsv, streamdataunit, streamdatetimeformat, and multiplicationfactor. A "RESPONSE SCHEMA (CLICK TO EXPAND)" section is also present.

This API can be accessed directly from the code in any language.  
Below is the code sample to get the list of streams for a data source:

```
API='your API key'  
SOURCE_KEY='source key of data source'  
# Get list of data streams for any data source  
def get_data_streams(self):  
    self.server_address = 'http://www.dataglen.net'  
    self.sources_url = '/api/sources/  
    self.api_key = API  
    self.token = 'Token ' + self.api_key  
    self.url = self.server_address + self.sources_url + SOURCE_KEY + '/streams/'  
    self.auth_header = {'Authorization': self.token}  
    response = requests.get(self.url, headers=self.auth_header)  
    print(response.status_code)
```

## 2.2.2 Create a new Data Stream

**API: /api/sources/{source\_key}/streams (POST)**

**Description:** You can add data streams for any data source using above API

**Parameters:**

Parameter Name	Parameter Type	Data Type
source_key	path	string
body	body	body

## Response:

**name (string):** Name of the data stream, should be unique for this source,  
**source (string):** Key of the source this stream belongs to,  
**streamDataType (choice) = ['INTEGER' or 'BOOLEAN' or 'STRING' or 'FLOAT' or 'LONG' or 'MAC' or 'TIMESTAMP' or 'DATE' or 'TIME']:** Type of the data this stream reports,  
**streamDataUnit (string):** Data Unit,  
**streamDateTimeFormat (string):** DateTime field values should be reported in ISO 8601 formats. You can specify custom formats here,  
**streamPositionInCSV (integer):** Applicable only if source dataFormat is CSV, should be unique,  
**multiplicationFactor (number):** Multiplication factor for this stream, applicable only for numerical streams. Defaults to one.

## Error Status Codes

HTTP Status Code	Reason
201	Success. Accompanied with details of the new stream.
400	If the source key mentioned is invalid (INVALID_SOURCE_KEY) or If any of the essential parameters are missing/invalid (BAD_REQUEST).
409	If a stream with the mentioned name OR position already exists (DUPLICATE_STREAM).
401	Not authenticated

This API looks like the below image on the API docs page:

This API can be accessed directly from the code in any language.  
Below is the code sample to add a data stream for any data source:

```
API='your API key'
SOURCE_KEY='source key of data source'
# Add a new stream for a data source
def add_data_stream(self):
    self.server_address = 'http://www.dataglen.net'
    self.sources_url = '/api/sources/'
    self.api_key = API
    self.token = 'Token ' + self.api_key
    self.url = self.server_address + self.sources_url + SOURCE_KEY + '/streams/'
    self.post_body = {
        "name": "test_stream2",
        "streamDataType": "INTEGER",
        "streamPositionInCSV": 2
```

```

        }
length = sys.getsizeof(self.post_body)
self.auth_header = {'Authorization': self.token, 'content-length': length, 'content-type': 'application/json'}
response = requests.post(self.url, data=json.dumps(self.post_body),
headers=self.auth_header)
print(response.status_code)

```

## 2.2.3 Delete a Data stream

**API:** /api/sources/{source\_key}/streams/{stream\_name} (DELETE)

**Description:** You can delete a data stream using above API.

**Parameters:**

Parameter Name	Parameter Type	Data Type
source_key	path	string
stream_name	path	string

### Error Status Codes

HTTP Status Code	Reason
204	Success. The source has been deleted.
400	If the specified source key does not exist (INVALID_SOURCE_KEY) or If the specified stream name does not exist (INVALID_DATA_STREAM).
401	Not authenticated

This API looks like the below image on the API docs page:

The screenshot shows a web browser displaying the API documentation for `dataglen.net`. The URL in the address bar is `dataglen.net/api/docs/#!/Action_Streams/Action_Streams_destroy`. The page is titled "Delete an action data stream". On the left, there's a sidebar titled "API REFERENCE" with a "TOKEN" input field containing a redacted token. The main content area has a "DELETE" button followed by the endpoint URL `/api/sources/{source_key}/action_streams/{action_stream_name}`. Below this, there's a "Parameters" section with two fields: "source\_key" (required, string) and "action\_stream\_name" (required, string). A "TRY" button is located at the bottom of the parameters section. To the right, there's a "RESPONSE SAMPLE" section showing a sample response of "0" and a "RESPONSE SCHEMA (CLICK TO EXPAND)" section which is currently collapsed.

## 2.2.4 Get a stream with mentioned name

**API: /api/sources/{source\_key}/streams/{stream\_name} (GET)**

**Description:** You can get the details of any particular stream using above API.

### Parameters:

Parameter Name	Parameter Type	Data Type
----------------	----------------	-----------

source_key	path	string
stream_name	path	string

**Response:**

**name (string):** Name of the data stream, should be unique for this source,  
**source (string):** Key of the source this stream belongs to,  
**streamDataType (choice) = ['INTEGER' or 'BOOLEAN' or 'STRING' or 'FLOAT' or 'LONG' or 'MAC' or 'TIMESTAMP' or 'DATE' or 'TIME']:** Type of the data this stream reports,  
**streamDataUnit (string):** Data Unit,  
**streamDateTimeFormat (string):** DateTime field values should be reported in ISO 8601 formats. You can specify custom formats here,  
**streamPositionInCSV (integer):** Applicable only if source dataFormat is CSV, should be unique,  
**multiplicationFactor (number):** Multiplication factor for this stream, applicable only for numerical streams. Defaults to one.

### Error Status Codes

HTTP Status Code	Reason
200	Success
400	If the specified source key does not exist (INVALID_SOURCE_KEY) or If the specified stream name does not exist (INVALID_DATA_STREAM).
401	Not authenticated

This API looks like the below image on the API docs page:

The screenshot shows the API documentation for the `/api/sources/{source_key}/action_streams/{action_stream_name}` endpoint. The main content area displays the following details:

- Title:** Get an action stream with the mentioned key
- Method:** GET
- URL:** /api/sources/{source\_key}/action\_streams/{action\_stream\_name}
- Description:** Manage your action streams. Get an action stream with the mentioned key.
- Parameters:**
  - `source_key` (required) string
  - `action_stream_name` (required) string
- Test this endpoint:** A button to test the endpoint.
- RESPONSE SAMPLE:**

```
{
  "name": "string",
  "source": "string",
  "streamDataType": "INTEGER",
  "streamPositionInCSV": 0,
  "streamDataUnit": "string",
  "streamDateFormat": "string",
  "type": "string"
}
```
- RESPONSE SCHEMA (CLICK TO EXPAND):** A link to expand the response schema.

## 2.2.5 Update an existing Data Stream

**API:** `/api/sources/{source_key}/streams/{stream_name}` (PATCH)

**Description:** You can update an existing data stream using above API.

### Parameters:

Parameter Name	Parameter Type	Data Type
<code>source_key</code>	path	string
<code>stream_name</code>	path	string
<code>body</code>	body	body

### Response:

**name (string)**: Name of the data stream, should be unique for this source,  
**source (string)**: Key of the source this stream belongs to,  
**streamDataType (choice) = ['INTEGER' or 'BOOLEAN' or 'STRING' or 'FLOAT' or 'LONG' or 'MAC' or 'TIMESTAMP' or 'DATE' or 'TIME']**: Type of the data this stream reports,  
**streamDataUnit (string)**: Data Unit,  
**streamDateTimeFormat (string)**: DateTime field values should be reported in ISO 8601 formats. You can specify custom formats here,  
**streamPositionInCSV (integer)**: Applicable only if source dataFormat is CSV, should be unique,  
**multiplicationFactor (number)**: Multiplication factor for this stream, applicable only for numerical streams. Defaults to one.

### Error Status Codes

HTTP Status Code	Reason
201	Success
401	Not authenticated
400	If the specified key does not exist (INVALID_SOURCE_KEY) or If the specified key does not exist (INVALID_DATA_STREAM).

This API looks like the below image on the API docs page:

**TOKEN**

**API REFERENCE**

- Data Sources
- Data Streams
  - Get a list of streams for a source
  - Create a new data stream
  - Delete a data stream
  - Get a stream with the mentioned ...
  - Update an existing data s...**
- Data Management
- Action Streams
- Action Management
- Config Streams
- Config Management
- Solar Plants
- Color Inverters

**Update an existing data stream**

**PATCH** /api/sources/{source\_key}/streams/{stream\_name}/

Manage your streams

Update an existing data stream.

Parameters

source_key	(required)
string	

stream_name	(required)
string	

body	
------	--

**RESPONSE SAMPLE**

```
{
  "name": "string",
  "source": "string",
  "streamdatatype": "INTEGER",
  "streamPositionInCSV": 0,
  "streamDataInit": "string",
  "streamDateTimeFormat": "string",
  "multiplicationFactor": 0
}
```

**RESPONSE SCHEMA (CLICK TO EXPAND)**

**BODY SAMPLE**

```
{
  "name": "string",
  "source": "string",
  "streamdatatype": "INTEGER",
  "streamPositionInCSV": 0,
  "streamDataInit": "string",
  "streamDateTimeFormat": "string",
  "multiplicationFactor": 0
}
```

**BODY SCHEMA (CLICK TO EXPAND)**

## 2.3 API's related to Data Management

### 2.3.1 Get a list of data records for a source and a set of data streams

**API: /api/sources/{source\_key}/data/ (GET)**

**Description:** You can get a list of data records for a source and a set of data streams specified using above API for a particular span of time specified. If the number of records is greater than 1000, then the latest 1000 records are fetched.

**Parameters:**

Parameter Name	Parameter Type	Description	Data Type
source_key	path		string
startTime	query	Start time of data lookup. It should be in ISO 8601 format. If there's no timezone mentioned, the default timezone will be of the source.	string
endTime	query	End time of data lookup. It should be in ISO 8601 format. If there's no timezone mentioned, the default timezone will be of the source.	string
streamNames	query	A comma separated list of stream names that should be included in the data. If this parameter is not specified, all streams will be included.	string

If no stream name is specified, it shows the records for all the data streams.

**Response:**

```
{  
  "sourceKey": "string",  
  "streams": [  
    {  
      "count": 0,  
      "records": [  
        {  
          "id": 1,  
          "value": "Hello World"  
        }  
      ]  
    }  
  ]  
}
```

```

    "endTime": "2016-01-12T16:25:44.917Z",
    "name": "string",
    "startTime": "2016-01-12T16:25:44.917Z",
    "timestamps": [
        "string"
    ],
    "values": [
        "string"
    ]
}
]
}

```

## Error Status Codes

HTTP Status Code	Reason
200	Success
400	If essential parameters are not specified in the request or dates are not mentioned in the correct format (BAD_REQUEST) or If the source key mentioned is invalid (INVALID_SOURCE_KEY) or If there is an invalid stream present in streamsNames (INVALID_DATA_STREAM)
401	Not authenticated

This API looks like the below image on the API docs page:

The screenshot shows the API documentation for the `Data_Records_Set_list` endpoint. The left sidebar lists various API reference categories. The main content area is titled "Get a list of data records for a source and a set of streams". It provides a `GET` method, a URL path `/api/sources/{source_key}/data/`, and a detailed description of the endpoint. It also shows the required parameters `source_key` and `startTime`, and a sample response in JSON format.

This API can be accessed directly from the code in any language.

Below is the code sample to fetch the data records in python. Using this code sample all the data records can be fetched within the time span specified:

```
API='your API key'
SOURCE_KEY='source key of data source'
#get data records
def getData(self,st,et):
    self.server_address = 'http://www.dataglen.net'
    self.sources_url = '/api/sources/'
    self.api_key = API
    self.token = 'Token ' + self.api_key
    streamNames = 'ADC1_R'
    count = 1000
    total_count = 0
    self.url = self.server_address + self.sources_url + SOURCE_KEY + '/data/'
    self.auth_header = {'Authorization ': self.token }
    while count==1000:
        self.params = {'startTime': str(st), 'endTime': str(et), 'streamNames':
str(streamNames)}
```

```

response = requests.get(self.url, headers=self.auth_header, params=self.params)
resp_dict = json.loads(response.content)
count = resp_dict['streams'][0]['count']
if count == 0:
    break
et = resp_dict['streams'][0]['startTime']
print(resp_dict['streams'][0]['count']),
(resp_dict['streams'][0]['startTime']), (resp_dict['streams'][0]['endTime'])

```

Above code snippet is written for one data stream which can be extended to any number of streams.

### 2.3.2 Write a new data record

**API:** `/api/sources/{source_key}/data/ (POST)`

**Description:** You can write new data records for the data streams of any source.

**Parameters:**

Parameter Name	Parameter Type	Data Type
source_key	path	string
body	body	body

#### Error Status Codes

HTTP Status Code	Reason
201	Success. New action has been written into the database.
400	If the source key mentioned is invalid (INVALID_SOURCE_KEY) or If any of the essential parameters are missing/invalid (BAD_REQUEST) or If there is an error

retrieving the payload (ERROR\_RETRIEVING\_PAYLOAD) or If there is an error splitting multiple records (ERROR\_SPLITTING\_RECORDS) or If there is an error in parsing multiple action streams (ERROR\_SPLITTING\_STREAMS) or If the data provided in the request body is not proper as per the data units of streams (INVALID\_DATA).

401

Not authenticated

This API looks like the below image on the API docs page:

The screenshot shows a web browser displaying the API documentation for `dataglen.net/api/docs/#!/Data_Management/Data_Records_Set_create`. The left sidebar contains a 'TOKEN' input field and a 'API REFERENCE' section with links to Data Sources, Data Streams, Data Management, Action Streams, Action Management, Config Streams, Config Management, Solar Plants, and Solar Inverters. A 'DATAGLEN' button is also present. The main content area is titled 'Write a new data record' and describes a POST request to `/api/sources/{source_key}/data/`. It says 'Manage your data records' and 'Write a new data record. Payload should be in the semantics as specified in the sensor configuration.' Below this is a 'Parameters' section with two fields: 'source\_key' (required, string) and 'body' (required). The 'body' field has a 'Content type:' dropdown set to 'text/plain'. To the right, there are sections for 'RESPONSE SAMPLE' (showing '0'), 'RESPONSE SCHEMA (CLICK TO EXPAND)', 'BODY SAMPLE', and 'BODY SCHEMA (CLICK TO EXPAND)'.

This API can be accessed directly from the code in any language.  
Below is the code sample to write new data record in CSV format.

```
API='your API key'  
SOURCE_KEY='source key of data source'  
# write a new data record in csv format
```

```

def write_data_csv(self):
    self.server_address = 'http://www.dataglen.net'
    self.sources_url = '/api/sources/'
    self.api_key = API
    self.token = 'Token ' + self.api_key
    self.url = self.server_address + self.sources_url + SOURCE_KEY+ '/data/'
    self.post_body = '1,2'
    length = sys.getsizeof(self.post_body)
    self.auth_header = {'Authorization ': self.token, 'content-length': length, 'content-type': 'text/plain'}
    response = requests.post(self.url, data=self.post_body, headers=self.auth_header)
    print(response.status_code)

```

Below is the code sample to write data records in json format:

```

API='your API key'
SOURCE_KEY='source key of data source'
# write a new data record in json format
def write_data_json(self):
    self.server_address = 'http://www.dataglen.net'
    self.sources_url = '/api/sources/'
    self.api_key = API
    self.token = 'Token ' + self.api_key
    self.url = self.server_address + self.sources_url + SOURCE_KEY+ '/data/'
    self.post_body = [{"test_stream1":100}, {"test_stream2":200}]
    length = sys.getsizeof(self.post_body)
    self.auth_header = {'Authorization ': self.token, 'content-length': length, 'content-type': 'application/json'}
    response = requests.post(self.url, data=json.dumps(self.post_body),
    headers=self.auth_header)
    print(response.status_code)

```

## 2.4 API's related to Action Management

### 2.4.1 Get a list of Pending actions for a given data source

**API:** /api/sources/{source\_key}/action/ (GET)

**Description:** You can get all the output fields for any source for which the state/value has been changed and the changes that are not acknowledged using above API i.e. the pending actions for a given source.

**Parameters:**

Parameter Name	Parameter Type	Data Type
source_key	path	string

```
{  
  "insertionTime": "2016-01-12T16:25:44.956Z",  
  "sourceKey": "string",  
  "streams": [  
    {  
      "name": "string",  
      "value": "string"  
    }  
  ]  
}
```

#### Error Status Codes

HTTP Status Code	Reason
200	Success

400 If the source key mentioned in invalid (INVALID\_SOURCE\_KEY) or If essential parameters are not specified in the request or dates are not mentioned in the correct format.(BAD\_REQUEST)

401 Not authenticated

This API looks like below image on the API docs page:

The screenshot shows a web browser displaying the DataGlen API documentation. The URL in the address bar is `dataglen.net/api/docs/#!/Action_Management/Action_Records_Set_list`. The left sidebar contains a 'TOKEN' input field and a 'API REFERENCE' section with links to various endpoints: Data Sources, Data Streams, Data Management, Action Streams, Action Management, and a 'Get a list of pending actions...' link which is currently selected. The main content area has a title 'Get a list of pending actions for a given source'. Below it is a 'GET /api/sources/{source\_key}/action/' endpoint description. A 'Parameters' section includes a 'source\_key' input field with '(required)' and 'string' annotations. A 'TRY' button is available for testing the endpoint. On the right, there is a 'RESPONSE SAMPLE' showing a JSON snippet and a 'RESPONSE SCHEMA (CLICK TO EXPAND)' section.

This API can be accessed directly from the code in any language. Below is the code sample to get a list of pending devices in python.

```
API='your API key'  
SOURCE_KEY='source key of data source'  
# get action  
def action_get(self):  
    self.server_address = 'http://www.dataglen.net'
```

```

self.sources_url = '/api/sources'
self.api_key = API
self.token = 'Token ' + self.api_key
self.url = self.server_address + self.sources_url + SOURCE_KEY + '/action/'
self.auth_header = {'Authorization': self.token}
response = requests.get(self.url, headers=self.auth_header)
print(response.status_code)

```

## 2.4.2 Write a new action record

### **API: /api/sources/{source\_key}/action/ (POST)**

**Description:** You can create a new action record using above API.

#### **Parameters:**

Parameter Name	Parameter Type	Data Type
source_key	path	string
body	body	body

#### **Error Status Codes**

HTTP Status Code	Reason
201	Success. New action has been written into the database.
400	If the source key mentioned is invalid (INVALID_SOURCE_KEY) or If any of the essential parameters are missing/invalid (BAD_REQUEST) or If there is an error retrieving the payload (ERROR_RETRIEVING_PAYLOAD) or If there is an error splitting multiple records (ERROR_SPLITTING_RECORDS) or If there is an error in

parsing multiple action streams (ERROR\_SPLITTING\_STREAMS) or If the data provided in the request body is not proper as per the data units of streams (INVALID\_DATA).

401 Not authenticated

This API looks like below image on the API docs page:

The screenshot shows a web browser displaying the API documentation for the `/api/sources/{source_key}/action/` endpoint. The left sidebar contains a **TOKEN** input field and a **API REFERENCE** section with links to various API endpoints. The main content area has a title **Write a new action record** and a **POST** method. It includes a description: "Write a new action record. Payload should be in the semantics as specified in the sensor configuration." Below this is a **Parameters** section with two fields: `source_key` (required, string) and `body` (required). The `body` field has a **Content type:** dropdown set to `application/json`. To the right, there are sections for **RESPONSE SAMPLE** (empty), **RESPONSE SCHEMA (CLICK TO EXPAND)**, **BODY SAMPLE** (empty), and **BODY SCHEMA (CLICK TO EXPAND)**.

This API can be accessed directly from the code in any language. Below is the code sample to get a list of pending devices in python.

```
API='your API key'  
SOURCE_KEY='source key of data source'  
# create action record
```

```
def action_create_post(self):
    self.server_address = 'http://www.dataglen.net'
    self.sources_url = '/api/sources/'
    self.api_key = API
    self.token = 'Token ' + self.api_key
    self.url = self.server_address + self.sources_url + SOURCE_KEY + '/action/'
    self.post_body = {"action-stream1":10,"action-stream2":20}
    length = sys.getsizeof(self.post_body)
    self.auth_header = {'Authorization ': self.token, 'content-length': length, 'content-type': 'application/json'}
    response = requests.post(self.url , data =json.dumps(self.post_body) ,
headers=self.auth_header)
    print(response.status_code)
```