



Onsight Flow Developer API Guide

Copyright

Librestream Onsight Flow API Guide

Doc #: 400382-01 Rev: A

February 2022 (v.2.25)

Information in this document is subject to change without notice. Reproduction in any manner whatsoever without the written permission of Librestream is strictly forbidden.

Copyright Notice:

Copyright 2004-2022 Librestream Technologies Incorporated. All Rights Reserved.

Patents Notice:

United States Patent # 7,221,386, together with additional patents pending in Canada, the United States, and other countries, all of which are in the name of Librestream Technologies Inc.

Trademark Notice

Librestream, the Librestream logo, Onsight, the Onsight logo, Onsight Connect, Onsight Flow, Onsight Workspace, Onsight Cube, Onsight Collaboration Hub, Onsight Smartcam, Onsight Platform Manager, and Onsight Teamlink are either registered trademarks or trademarks of Librestream Technologies Incorporated in Canada, the United States, European Union and/or other countries. All other trademarks are the property of their respective owners.

Contents

Copyright.....	ii
1. Onsight Flow Developer Documentation.....	19
2. Authentication.....	21
2.1. OAuth2.....	21
2.1.1. 1. Accessing a token.....	21
2.1.2. 2. Make an API Request.....	21
2.2. Permissions, Scopes & Policies.....	22
2.2.1. Policies.....	22
2.2.2. Permissions.....	22
2.2.3. Scopes.....	22
2.2.4. Example.....	22
2.3. Create an M2M Client.....	22
2.3.1. 1. Get a Token.....	22
2.3.2. 2. Decide upon the Scopes and Permissions for your Client.....	23
2.3.3. 3. Request a new Client.....	24
2.3.4. 4. Use the Client to request Tokens.....	25
2.4. Policies.....	25
2.4.1. Client_Admin.....	25
2.4.2. Job_Modify.....	25
2.4.3. Job_Read.....	26
2.4.4. Job_ReadAll.....	26
2.4.5. Query_Admin.....	26
2.4.6. Query_Read.....	27
2.4.7. Reports_Admin.....	27
2.4.8. Reports_Create.....	27
2.4.9. Reports_Modify.....	28
2.4.10. Reports_Read.....	28
2.4.11. Schedule_Admin.....	28
2.4.12. Schedule_Read.....	29
2.4.13. Schedule_ReadAll.....	29
2.4.14. Spoke_Client.....	29
2.4.15. Team_Admin.....	30
2.4.16. TeamSettings_Admin.....	30
2.4.17. TeamSettings_Read.....	30

2.4.18. Trigger_Admin.....	30
2.4.19. Trigger_Execute.....	31
2.4.20. Trigger_Read.....	31
2.4.21. Workflow_Admin.....	31
2.4.22. Workflow_Approve.....	32
2.4.23. Workflow_Create.....	32
2.4.24. Workflow_Delete.....	32
2.4.25. Workflow_Execute.....	33
2.4.26. Workflow_Modify.....	33
2.4.27. Workflow_Read.....	33
3. Query API.....	35
3.1. GraphQL Overview.....	35
3.1.1. Advantages compared to GET Requests.....	35
3.1.2. Trying out GraphQL.....	36
3.2. Getting Started.....	39
3.3. Getting API Access.....	39
3.3.1. Accessing Onsight Flow GraphQL API.....	40
3.3.2. Primary/Secondary Keys.....	40
3.3.3. Temporary Keys.....	41
3.3.4. Access Tokens.....	42
3.4. Query Designer.....	43
3.4.1. Writing Queries.....	44
3.4.2. Exporting Queries.....	44
3.4.3. CSV Queries.....	45
3.5. Querying Jobs.....	46
3.5.1. Introduction.....	46
3.5.2. Filtering on specific Jobs.....	47
3.5.3. Selecting Job Fields.....	52
3.5.4. Select Job Attachments.....	55
3.6. Ingest Data to Excel	57
3.6.1. 1. Create the Query.....	58
3.6.2. 2. Select the Output Format.....	60
3.6.3. 3. Execute Query from Excel.....	60
3.6.4. 4. Manage the Query.....	62
3.6.5. A note on Power BI.....	63
3.7. Example Queries.....	63

3.7.1. Get Latest Jobs.....	64
3.7.2. Get All Reports.....	64
3.7.3. Get All Attachments (original file only).....	64
3.7.4. Get All Attachment Thumbnails.....	65
3.7.5. Get Substep Tags.....	65
3.7.6. Get Job Level Data.....	66
3.7.7. Get All Users.....	67
4. Collections.....	69
4.1. Get Collections.....	70
4.1.1. HTTP Request.....	70
4.1.2. Authorization.....	70
4.1.3. Headers.....	70
4.1.4. Response Body.....	71
4.2. Update Collection Name.....	71
4.2.1. HTTP Request.....	71
4.2.2. Authorization.....	71
4.2.3. Headers.....	72
4.2.4. Path Parameters.....	72
4.2.5. Request Body.....	72
4.3. Create Collection.....	72
4.3.1. HTTP Request.....	72
4.3.2. Authorization.....	72
4.3.3. Headers.....	72
4.3.4. Request Body.....	72
4.3.5. Response Body.....	73
4.4. Delete Collection.....	73
4.4.1. HTTP Request.....	73
4.4.2. Authorization.....	73
4.4.3. Headers.....	73
4.4.4. Path Parameters.....	73
4.5. Upload Collection.....	73
4.5.1. Overview.....	73
4.5.2. Notes.....	74
4.5.3. Create Collection Upload Session.....	74
4.5.4. Upload Collection File.....	75

5. Groups.....	77
5.1. Get All Groups.....	77
5.1.1. HTTP Request.....	77
5.1.2. Authorization.....	77
5.1.3. Headers.....	77
5.1.4. Response Body.....	77
5.2. Create Group.....	78
5.2.1. HTTP Request.....	78
5.2.2. Authorization.....	78
5.2.3. Headers.....	78
5.2.4. Request Body.....	78
5.2.5. Response Body.....	78
5.3. Get Group Details.....	78
5.3.1. HTTP Request.....	78
5.3.2. Authorization.....	79
5.3.3. Headers.....	79
5.3.4. Path Parameters.....	79
5.3.5. Response Body.....	79
5.4. Update Group Name.....	79
5.4.1. HTTP Request.....	79
5.4.2. Authorization.....	79
5.4.3. Headers.....	80
5.4.4. Path Parameters.....	80
5.4.5. Request Body.....	80
5.4.6. Response Body.....	80
5.5. Update Group Permissions.....	80
5.5.1. HTTP Request.....	80
5.5.2. Authorization.....	80
5.5.3. Headers.....	81
5.5.4. Path Parameters.....	81
5.5.5. Request Body.....	81
5.6. Add Users to Group.....	81
5.6.1. HTTP Request.....	81
5.6.2. Authorization.....	81
5.6.3. Headers.....	81
5.6.4. Path Parameters.....	82

5.6.5. Request Body.....	82
5.7. Remove Users from Group.....	82
5.7.1. HTTP Request.....	82
5.7.2. Authorization.....	82
5.7.3. Headers.....	82
5.7.4. Path Parameters.....	82
5.7.5. Request Body.....	82
5.8. Delete Group.....	83
5.8.1. HTTP Request.....	83
5.8.2. Authorization.....	83
5.8.3. Headers.....	83
5.8.4. Path Parameters.....	83
6. Jobs.....	85
6.1. Get All Jobs.....	85
6.1.1. HTTP Request.....	85
6.1.2. Authorization.....	85
6.1.3. Headers.....	85
6.1.4. Query Parameters.....	85
6.1.5. Response Body.....	86
6.2. Get Jobs by Workflow Id.....	87
6.2.1. HTTP Request.....	87
6.2.2. Authorization.....	87
6.2.3. Headers.....	87
6.2.4. Path Parameters.....	87
6.2.5. Response Body.....	87
6.3. Get Snapshots of a Job.....	89
6.3.1. HTTP Request.....	89
6.3.2. Authorization.....	89
6.3.3. Headers.....	90
6.3.4. Path Parameters.....	90
6.3.5. Response Body.....	90
6.4. Get Job by Client Job Id.....	90
6.4.1. HTTP Request.....	90
6.4.2. Authorization.....	90
6.4.3. Headers.....	90
6.4.4. Path Parameters.....	90

6.4.5. Response Body.....	90
6.5. Get Job by Job Id.....	92
6.5.1. HTTP Request.....	92
6.5.2. Authorization.....	92
6.5.3. Headers.....	92
6.5.4. Path Parameters.....	92
6.5.5. Response Body.....	92
6.6. Get My Jobs.....	94
6.6.1. HTTP Request.....	94
6.6.2. Authorization.....	94
6.6.3. Headers.....	94
6.6.4. Query Parameters.....	95
6.6.5. Response Body.....	95
6.7. Abort a Job.....	96
6.7.1. HTTP Request.....	96
6.7.2. Authorization.....	96
6.7.3. Headers.....	96
6.7.4. Path Parameters.....	97
6.8. Delete a Job.....	97
6.8.1. HTTP Request.....	97
6.8.2. Authorization.....	97
6.8.3. Headers.....	97
6.8.4. Path Parameters.....	97
6.9. Assign User to a Job.....	97
6.9.1. HTTP Request.....	97
6.9.2. Authorization.....	97
6.9.3. Headers.....	98
6.9.4. Path Parameters.....	98
6.10. Exclude a Job.....	98
6.10.1. HTTP Request.....	98
6.10.2. Authorization.....	98
6.10.3. Headers.....	98
6.10.4. Path Parameters.....	98
6.10.5. Request Body.....	98
6.11. Add Metadata to a Job.....	99
6.11.1. HTTP Request.....	99

6.11.2. Authorization.....	99
6.11.3. Headers.....	99
6.11.4. Path Parameters.....	99
6.11.5. Request Body.....	99
6.12. Create a Job.....	99
6.12.1. HTTP Request.....	100
6.12.2. Authorization.....	100
6.12.3. Headers.....	100
6.12.4. Request Body.....	100
6.12.5. Response Body.....	101
6.13. Abort Jobs for an User.....	102
6.13.1. HTTP Request.....	102
6.13.2. Authorization.....	102
6.13.3. Headers.....	103
6.13.4. Query Parameters.....	103
6.14. Basic Sync.....	103
6.14.1. Overview.....	103
6.14.2. HTTP Request.....	103
6.14.3. Authorization.....	103
6.14.4. Headers.....	103
6.14.5. Request Body.....	103
6.15. Update a Job.....	105
6.15.1. HTTP Request.....	105
6.15.2. Authorization.....	105
6.15.3. Headers.....	105
6.15.4. Path Parameters.....	105
6.15.5. Request Body.....	105
7. Report Generator.....	107
7.1. Generate a Report.....	107
7.1.1. HTTP Request.....	107
7.1.2. Authorization.....	107
7.1.3. Headers.....	107
7.1.4. Query Parameters.....	107
7.1.5. Response Body.....	107
7.2. Generate Report by Report Template Id.....	108
7.2.1. HTTP Request.....	108

7.2.2. Authorization.....	108
7.2.3. Headers.....	108
7.2.4. Query Parameters.....	108
7.2.5. Path Parameters.....	108
7.2.6. Response Body.....	108
8. Reports.....	109
8.1. Get PDF by Client Job Id.....	109
8.1.1. HTTP Request.....	109
8.1.2. Authorization.....	109
8.1.3. Headers.....	109
8.1.4. Query Parameters-1234.....	109
8.1.5. Path Parameters.....	109
8.1.6. Response Body.....	109
9. Report Templates.....	111
9.1. Get All Report Templates.....	111
9.1.1. HTTP Request.....	111
9.1.2. Authorization.....	111
9.1.3. Headers.....	111
9.1.4. Response Body.....	111
9.2. Get Report Template.....	113
9.2.1. HTTP Request.....	113
9.2.2. Authorization.....	113
9.2.3. Headers.....	113
9.2.4. Path Parameters.....	113
9.2.5. Response Body.....	113
9.3. Get Report Template Names.....	114
9.3.1. HTTP Request.....	114
9.3.2. Authorization.....	114
9.3.3. Headers.....	114
9.3.4. Response Body.....	114
9.4. Get Default Template.....	115
9.4.1. HTTP Request.....	115
9.4.2. Authorization.....	115
9.4.3. Headers.....	115
9.5. Delete Report Template.....	115
9.5.1. HTTP Request.....	116

9.5.2. Authorization.....	116
9.5.3. Headers.....	116
9.5.4. Path Parameters.....	116
9.6. Set Default Report Template.....	116
9.6.1. HTTP Request.....	116
9.6.2. Authorization.....	116
9.6.3. Headers.....	116
9.6.4. Path Parameters.....	116
9.7. Unset Default Report Template.....	117
9.7.1. HTTP Request.....	117
9.7.2. Authorization.....	117
9.7.3. Headers.....	117
9.8. Create Report Template.....	117
9.8.1. HTTP Request.....	117
9.8.2. Authorization.....	117
9.8.3. Headers.....	117
9.8.4. Request Body.....	117
9.8.5. Response Body.....	118
9.9. Update Template Metadata.....	118
9.9.1. HTTP Request.....	118
9.9.2. Authorization.....	118
9.9.3. Headers.....	119
9.9.4. Path Parameters.....	119
9.9.5. Request Body.....	119
9.10. Update Report Template.....	119
9.10.1. HTTP Request.....	119
9.10.2. Authorization.....	119
9.10.3. Headers.....	119
9.10.4. Path Parameters.....	120
9.10.5. Request Body.....	120
10. Scheduler.....	121
10.1. Get All Scheduled Jobs.....	121
10.1.1. HTTP Request.....	121
10.1.2. Authorization.....	121
10.1.3. Headers.....	121
10.1.4. Response Body.....	121

10.2. Get Scheduled Job.....	123
10.2.1. HTTP Request.....	123
10.2.2. Authorization.....	123
10.2.3. Headers.....	123
10.2.4. Path Parameters.....	123
10.2.5. Response Body.....	123
10.3. Get My Scheduled Jobs.....	124
10.3.1. HTTP Request.....	124
10.3.2. Authorization.....	124
10.3.3. Headers.....	125
10.3.4. Response Body.....	125
10.4. Update the status of a Task in a Sheduled Job.....	125
10.4.1. HTTP Request.....	125
10.4.2. Authorization.....	125
10.4.3. Headers.....	126
10.4.4. Path Parameters.....	126
10.4.5. Request Body.....	126
10.5. Update the Status of a Scheduled Job.....	126
10.5.1. HTTP Request.....	126
10.5.2. Authorization.....	126
10.5.3. Headers.....	126
10.5.4. Path Parameters.....	127
10.5.5. Request Body.....	127
10.6. Update Assigned User for a Job.....	127
10.6.1. HTTP Request.....	127
10.6.2. Authorization.....	127
10.6.3. Headers.....	127
10.6.4. Path Parameters.....	127
10.6.5. Request Body.....	127
10.7. Delete a Scheduled Job.....	128
10.7.1. HTTP Request.....	128
10.7.2. Authorization.....	128
10.7.3. Headers.....	128
10.7.4. Path Parameters.....	128
10.8. Create a Scheduled Job.....	128
10.8.1. HTTP Request.....	128

10.8.2. Authorization.....	128
10.8.3. Headers.....	129
10.8.4. Request Body.....	129
10.8.5. Response Body.....	129
11. TeamSettings.....	131
11.1. Set TeamSettings to Defaults.....	131
11.1.1. HTTP Request.....	131
11.1.2. Authorization.....	131
11.1.3. Headers.....	131
11.2. Set TeamSettings.....	131
11.2.1. HTTP Request.....	131
11.2.2. Authorization.....	131
11.2.3. Headers.....	132
11.2.4. Request Body.....	132
11.3. Get TeamSetting.....	132
11.3.1. HTTP Request.....	132
11.3.2. Authorization.....	132
11.3.3. Headers.....	132
11.3.4. Path Parameters.....	132
11.3.5. Response Body.....	132
11.4. Get TeamSettings.....	133
11.4.1. HTTP Request.....	133
11.4.2. Authorization.....	133
11.4.3. Headers.....	133
11.4.4. Response Body.....	133
12. Trigger.....	135
12.1. Get All Triggers.....	135
12.1.1. HTTP Request.....	135
12.1.2. Authorization.....	135
12.1.3. Headers.....	135
12.1.4. Response Body.....	135
12.2. Get Trigger by Trigger Id.....	136
12.2.1. HTTP Request.....	136
12.2.2. Authorization.....	136
12.2.3. Headers.....	137
12.2.4. Path Parameters.....	137

12.2.5. Response Body.....	137
12.3. Create Trigger.....	137
12.3.1. HTTP Request.....	137
12.3.2. Authorization.....	138
12.3.3. Headers.....	138
12.3.4. Request Body.....	138
12.3.5. Response Body.....	138
12.4. Update Trigger.....	138
12.4.1. HTTP Request.....	139
12.4.2. Authorization.....	139
12.4.3. Headers.....	139
12.4.4. Path Parameters.....	139
12.4.5. Request Body.....	139
12.5. Archive Trigger.....	139
12.5.1. HTTP Request.....	139
12.5.2. Authorization.....	140
12.5.3. Headers.....	140
12.5.4. Path Parameters.....	140
12.6. Unarchive Trigger.....	140
12.6.1. HTTP Request.....	140
12.6.2. Authorization.....	140
12.6.3. Headers.....	140
12.6.4. Path Parameters.....	140
13. Users.....	141
13.1. Get Users.....	141
13.1.1. HTTP Request.....	141
13.1.2. Authorization.....	141
13.1.3. Headers.....	141
13.1.4. Response Body.....	141
13.2. Get User.....	142
13.2.1. HTTP Request.....	142
13.2.2. Authorization.....	142
13.2.3. Headers.....	142
13.2.4. Path Parameters.....	142
13.2.5. Response Body.....	142
13.3. Create User.....	143

13.3.1. HTTP Request.....	143
13.3.2. Authorization.....	143
13.3.3. Headers.....	143
13.3.4. Request Body.....	143
13.3.5. Response Body.....	143
13.4. Get User Permissions.....	144
13.4.1. HTTP Request.....	144
13.4.2. Authorization.....	144
13.4.3. Headers.....	144
13.4.4. Path Parameters.....	144
13.4.5. Response Body.....	144
13.5. Lock User.....	145
13.5.1. HTTP Request.....	145
13.5.2. Authorization.....	145
13.5.3. Headers.....	145
13.5.4. Path Parameters.....	145
13.6. Reset User's Password.....	145
13.6.1. HTTP Request.....	145
13.6.2. Authorization.....	145
13.6.3. Headers.....	145
13.6.4. Path Parameters.....	146
13.6.5. Response Body.....	146
13.7. Unlock User.....	146
13.7.1. HTTP Request.....	146
13.7.2. Authorization.....	146
13.7.3. Headers.....	146
13.7.4. Path Parameters.....	146
13.8. Update User Details.....	146
13.8.1. HTTP Request.....	147
13.8.2. Authorization.....	147
13.8.3. Headers.....	147
13.8.4. Path Parameters.....	147
13.8.5. Request Body.....	147
13.9. Update User Permissions.....	147
13.9.1. HTTP Request.....	147
13.9.2. Authorization.....	147

13.9.3. Headers.....	148
13.9.4. Path Parameters.....	148
13.9.5. Request Body.....	148
13.10. Delete User.....	148
13.10.1. HTTP Request.....	148
13.10.2. Authorization.....	148
13.10.3. Headers.....	148
13.10.4. Path Parameters.....	149
14. Workflows.....	151
14.1. Get Workflows.....	151
14.1.1. HTTP Request.....	151
14.1.2. Authorization.....	151
14.1.3. Headers.....	151
14.1.4. Response Body.....	151
14.2. Get Workflow.....	152
14.2.1. HTTP Request.....	152
14.2.2. Authorization.....	152
14.2.3. Headers.....	153
14.2.4. Path Parameters.....	153
14.2.5. Response Body.....	153
14.3. Update Workflow Metadata.....	153
14.3.1. HTTP Request.....	154
14.3.2. Authorization.....	154
14.3.3. Headers.....	154
14.3.4. Path Parameters.....	154
14.3.5. Request Body.....	154
14.4. Upload a Workflow.....	154
14.4.1. Overview.....	155
14.4.2. Example.....	155
14.4.3. Create Workflow Upload Session.....	156
14.4.4. Upload Workflow Block.....	157
14.4.5. Finalize Wordflow Upload.....	158
14.5. Set Active Version.....	159
14.5.1. HTTP Request.....	159
14.5.2. Authorization.....	159
14.5.3. Headers.....	159

14.5.4. Path Parameters.....	159
14.6. Delete Workflow Version.....	160
14.6.1. HTTP Request.....	160
14.6.2. Authorization.....	160
14.6.3. Headers.....	160
14.6.4. Path Parameters.....	160
14.7. Delete Workflow.....	160
14.7.1. HTTP Request.....	160
14.7.2. Authorization.....	160
14.7.3. Headers.....	160
14.7.4. Path Parameters.....	161

1. Onsight Flow Developer Documentation

This website contains Developer Documentation for Onsight Flow as a digital Instruction Toolkit. This documentation will describe how to access data held within Onsight Flow programatically using our most common APIs.

Please see for the standard User Documentation.

2. Authentication

Related information

[OAuth2 \(on page 21\)](#)

[Permissions, Scopes & Policies \(on page 22\)](#)

[Create an M2M Client \(on page 22\)](#)

[Policies \(on page 25\)](#)

2.1. OAuth2

Onsight Flow uses OAuth2 exclusively* for authentication across all of its APIs.

More details as to the approaches and advantages that make OAuth2 a very good option for authentication can be found online, the primary source being the [IETFs own memo](#).

A developer or systems integrator working on an integration to Onsight Flow will need to follow a 2 stage process in order to access Onsight Flow APIs

1. Request or load a token
2. Make a request to an API using that token

2.1.1. 1. Accessing a token

Tokens can be requested from the Identity Server; in order to gain a token you must provide valid credentials. Once signed in to Onsight Flow you can request a token from the [Developer Tools page](#), however, manually requesting a token is not a useful approach for enabling an integration - you will want your integration to be able to request tokens from an API as they are needed.

The good news is that that API exists, however, for security reasons you cannot practicably access that API using your user credentials. Instead the recommended approach is to create a *machine to machine (M2M) client*, these clients are very similar to a user, they are enabled with one or more permissions that define what they are and aren't allowed to do, however, whilst a user in Onsight Flow represents a person, an M2M client represents a tool/product/integration that is permitted to interface with Onsight Flow.

2.1.2. 2. Make an API Request

Once you have either generated a new token or loaded an existing token you'll need to include it in your request within a header of the form

```
{
  "Authorization": "Bearer {token}"
}
```

* With the exception of the Query API, more information can be found in the Query API section

2.2. Permissions, Scopes & Policies

Three concepts need to be outlined in order to understand whether a request will be authorized.

2.2.1. Policies

Each API will have a Policy defined against it, in order to be authorized a request will have to meet the requirements of that Policy; a Policy includes a list of Scopes and Permissions, in order to meet the requirements of that Policy the request will need to include a match to at least one of the Scopes and one of the Permissions.

2.2.2. Permissions

Permissions reside with the user or client, they define the level of and types of action that the user/client can take along with the data that they can access

2.2.3. Scopes

Scopes reside with the token, they define the level of and types of action that the user/client can take along with the data that they can access

2.2.4. Example

Consider an application that primarily reads and reports data for a given service but occasionally is required to modify data on that service; in order to perform the full range of actions the client created for this application would probably need a high level Permission such as ADMIN, however the developer of the integration may decide to request and cache a token that includes only the <https://www.flow.librestream.com/auth.read> Scope in order to read the data whenever required and then on the occasions where data modification is required a one-off token could be generated that includes the <https://www.flow.librestream.com/auth/xxx.modify> Scope. This follows the principle of least privilege, a key principle for secure systems.

2.3. Create an M2M Client

2.3.1. 1. Get a Token

In order to create a machine to machine client you will need a token, that token needs to have been created by an OWNER user¹ and it needs to include one of the scopes that meet the [Client_Admin \(on page 25\)](#) policy.

¹ Ask your team administrators for the identity of your team owner(s), they are generally the people responsible for the account

Policies
?

Select required policies and Generate Access Token

☒ Client_Admin	☐ Job_Modify	☐ Job_Read
☐ Job_ReadAll	☐ Query_Admin	☐ Query_Read
☐ Reports_Admin	☐ Reports_Create	☐ Reports_Modify
☐ Reports_Read	☐ Schedule_Admin	☐ Schedule_Read
☐ Schedule_ReadAll	☐ Spoke_Client	☐ Team_Admin
☐ TeamSettings_Admin	☐ TeamSettings_Read	☐ Trigger_Admin
☐ Trigger_Execute	☐ Trigger_Read	☐ Workflow_Admin
☐ Workflow_Approve	☐ Workflow_Create	☐ Workflow_Delete
☐ Workflow_Execute	☐ Workflow_Modify	☐ Workflow_Read

GENERATE ACCESS TOKEN

Figure 2-1 Create a Client_Admin token

2.3.2. 2. Decide upon the Scopes and Permissions for your Client

In order to understand the scopes and permissions required by your client you'll first need to describe the range of actions that you want it to be able to take, for example

I want my application to be able to create scheduled jobs in Onsight Flow in order to do that it needs to not only be able to create the scheduled jobs but also access the list of workflows and users in order to assign that job to the right person and assign the relevant workflow

From this type of description you can then list out the APIs that you'll need to make requests to, in this case

- [Get Users](#)
- [Get Workflows](#)
- [Create Scheduled Job](#)

And by reviewing the policies for each of those APIs, in order to satisfy all policies I can see that I need the following permissions:

- ADMIN

And the following scopes:

- <https://www.flow.librestream.com/auth/workflow.read>
- <https://www.flow.librestream.com/auth/schedule.admin>

2.3.3. 3. Request a new Client

I can now combine the details together into a request to create the new client

```
curl --location --request POST 'https://accounts.flow.librestream.com/api/client/' \
--header 'Authorization: Bearer tokenGoesHere' \
--header 'Content-Type: application/json' \
--data-raw '{
  "name": "My Job Scheduling Application",
  "scopes": ["https://www.flow.librestream.com/auth/workflow.read",
"https://www.flow.librestream.com/auth/schedule.admin"],
  "permissions": ["admin"]
}'
```

And the response should be of the form

```
{
  "client": {
    "clientId": "6169571beee5ee0101a4b58c",
    "grantTypes": [
      "client_credentials"
    ],
    "secrets": [
      {
        "secretId": "4897852beee5ee0001a4b58b"
      }
    ],
    "teamName": "myteam",
    "displayName": "My Job Scheduling Application",
    "scopes": [
      "https://www.flow.librestream.com/auth/workflow.read",
      "https://www.flow.librestream.com/auth/schedule.admin"
    ],
    "allowAccessTokensViaBrowser": false,
    "clientPermissions": [
      "admin"
    ]
  },
  "newSecret": "8JnEe4471tvwc0GvsS0CUxpv1TW35CJJ"
}
```

As soon as you get this response, securely record both the clientId and the newSecret, this is your only chance to capture the newSecret for this client

2.4.2.1. Scopes

- <https://www.flow.librestream.com/auth/job.modify>
- <https://www.flow.librestream.com/auth/job.admin>

2.4.2.2. Permissions

- OWNER

2.4.3. Job_Read

2.4.3.1. Scopes

- <https://www.flow.librestream.com/auth/job.read>
- https://www.flow.librestream.com/auth/job.read_all
- <https://www.flow.librestream.com/auth/job.admin>

2.4.3.2. Permissions

- OWNER
- ADMIN

2.4.4. Job_ReadAll

2.4.4.1. Scopes

- https://www.flow.librestream.com/auth/job.read_all
- <https://www.flow.librestream.com/auth/job.modify>
- <https://www.flow.librestream.com/auth/job.admin>

2.4.4.2. Permissions

- OWNER
- ADMIN

2.4.5. Query_Admin

2.4.5.1. Scopes

- <https://www.flow.librestream.com/auth/query.admin>

2.4.5.2. Permissions

- OWNER
- ADMIN

2.4.6. Query_Read

2.4.6.1. Scopes

- <https://www.flow.librestream.com/auth/query.read>
- <https://www.flow.librestream.com/auth/query.admin>

2.4.6.2. Permissions

- OWNER
- ADMIN

2.4.7. Reports_Admin

2.4.7.1. Scopes

- <https://www.flow.librestream.com/auth/reports.admin>

2.4.7.2. Permissions

- OWNER
- ADMIN

2.4.8. Reports_Create

2.4.8.1. Scopes

- <https://www.flow.librestream.com/auth/reports.create>
- <https://www.flow.librestream.com/auth/reports.admin>

2.4.8.2. Permissions

- OWNER
- ADMIN
- EDITOR

2.4.9. Reports_Modify

2.4.9.1. Scopes

- <https://www.flow.librestream.com/auth//reports.modify>
- <https://www.flow.librestream.com/auth//reports.admin>

2.4.9.2. Permissions

- OWNER
- ADMIN

2.4.10. Reports_Read

2.4.10.1. Scopes

- <https://www.flow.librestream.com/auth/reports.read>
- <https://www.flow.librestream.com/auth/reports.create>
- <https://www.flow.librestream.com/auth/reports.admin>

2.4.10.2. Permissions

- OWNER
- ADMIN
- EDITOR

2.4.11. Schedule_Admin

2.4.11.1. Scopes

- <https://www.flow.librestream.com/auth/schedule.admin>

2.4.11.2. Permissions

- OWNER
- ADMIN

2.4.12. Schedule_Read

2.4.12.1. Scopes

- <https://www.flow.librestream.com/auth/schedule.read>
- https://www.flow.librestream.com/auth/schedule.read_all
- <https://www.flow.librestream.com/auth/schedule.admin>

2.4.12.2. Permissions

- OWNER
- ADMIN

2.4.13. Schedule_ReadAll

2.4.13.1. Scopes

- https://www.flow.librestream.com/auth/schedule.read_all
- <https://www.flow.librestream.com/auth/schedule.admin>

2.4.13.2. Permissions

- OWNER
- ADMIN

2.4.14. Spoke_Client

2.4.14.1. Scopes

- <https://www.flow.librestream.com/auth/spoke.client>

2.4.14.2. Permissions

- SPOKE

2.4.15. Team_Admin

2.4.15.1. Scopes

- <https://www.flow.librestream.com/auth//team.admin>

2.4.15.2. Permissions

- OWNER
- ADMIN

2.4.16. TeamSettings_Admin

2.4.16.1. Scopes

- <https://www.flow.librestream.com/auth/teamsettings.admin>

2.4.16.2. Permissions

- OWNER
- ADMIN

2.4.17. TeamSettings_Read

2.4.17.1. Scopes

- <https://www.flow.librestream.com/auth/teamsettings.read>
- <https://www.flow.librestream.com/auth/teamsettings.admin>

2.4.17.2. Permissions

- OWNER
- ADMIN
- EDITOR
- APPROVER

2.4.18. Trigger_Admin

2.4.18.1. Scopes

- <https://www.flow.librestream.com/auth/trigger.admin>

2.4.18.2. Permissions

- OWNER
- ADMIN

2.4.19. Trigger_Execute

2.4.19.1. Scopes

- <https://www.flow.librestream.com/auth/workflow.execute>
- <https://www.flow.librestream.com/auth/workflow.admin>

2.4.19.2. Permissions

- OWNER
- ADMIN

2.4.20. Trigger_Read

2.4.20.1. Scopes

- <https://www.flow.librestream.com/auth/trigger.read>
- <https://www.flow.librestream.com/auth/trigger.admin>

2.4.20.2. Permissions

- OWNER
- ADMIN

2.4.21. Workflow_Admin

2.4.21.1. Scopes

- <https://www.flow.librestream.com/auth/workflow.admin>

2.4.21.2. Permissions

- OWNER
- ADMIN

2.4.22. Workflow_Approve

2.4.22.1. Scopes

- <https://www.flow.librestream.com/auth/workflow.approve>

2.4.22.2. Permissions

- APPROVER

2.4.23. Workflow_Create

2.4.23.1. Scopes

- <https://www.flow.librestream.com/auth/workflow.create>
- <https://www.flow.librestream.com/auth/workflow.admin>

2.4.23.2. Permissions

- OWNER
- ADMIN
- EDITOR

2.4.24. Workflow_Delete

2.4.24.1. Scopes

- <https://www.flow.librestream.com/auth/workflow.delete>
- <https://www.flow.librestream.com/auth/workflow.admin>

2.4.24.2. Permissions

- OWNER
- ADMIN
- EDITOR

2.4.25. Workflow_Execute

2.4.25.1. Scopes

- <https://www.flow.librestream.com/auth/workflow.execute>

2.4.25.2. Permissions

2.4.26. Workflow_Modify

2.4.26.1. Scopes

- <https://www.flow.librestream.com/auth/workflow.modify>
- <https://www.flow.librestream.com/auth/workflow.admin>

2.4.26.2. Permissions

- OWNER
- ADMIN
- EDITOR

2.4.27. Workflow_Read

2.4.27.1. Scopes

- <https://www.flow.librestream.com/auth/workflow.read>
- <https://www.flow.librestream.com/auth/workflow.execute>
- <https://www.flow.librestream.com/auth/workflow.admin>
- <https://www.flow.librestream.com/auth/spoke.client>

2.4.27.2. Permissions

- OWNER
- ADMIN
- EDITOR
- APPROVER
- SPOKE

3. Query API

Related information

[GraphQL Overview \(on page 35\)](#)

[Getting Started \(on page 39\)](#)

[Getting API Access \(on page 39\)](#)

[Query Designer \(on page 43\)](#)

[Querying Jobs \(on page 46\)](#)

[Ingest Data to Excel \(on page 57\)](#)

[Example Queries \(on page 63\)](#)

3.1. GraphQL Overview

Onsight Flow surfaces all of its data through GraphQL, a technology originally developed within Facebook. GraphQL provides an API Surface for clients to query and a Runtime to execute those queries.

GraphQL is used instead of making traditional HTTP GET requests to grant API clients access to the underlying data. With this new technology, the client will construct a GraphQL query to be executed by the runtime. The fields requested will reflect the data returned.

Related information

[Advantages compared to GET Requests \(on page 35\)](#)

[More Information \(on page 36\)](#)

[Trying out GraphQL \(on page 36\)](#)

[Onsight Flow Query Designer \(on page 36\)](#)

[Postman \(on page 37\)](#)

[GET / POST Requests \(on page 38\)](#)

3.1.1. Advantages compared to GET Requests

Concise Queries

With a traditional REST API, the GET endpoints usually return all of the fields to satisfy all commands. This can be costly and usually returns data that the client doesn't need. With GraphQL, the clients (mobile app, web page, background processor) can specify only the fields that they need.

Database Abstraction

The clients do not need to understand the internal database schema or how the data is assembled. The GraphQL schema will define the data that can be queried without concerning the client as to how that data is constructed.

Accessibility

GraphQL queries are executed using standard HTTP GET or HTTP POST requests. A query can either be added to the end of a GET request, or, serialized into JSON and supplied in the body of a HTTP POST request.

Schema Additions

GraphQL Vendors (ie: Onsight Flow) can easily amend the schemas by adding extra fields. This will not increase the amount of data returned by existing clients, unless those clients choose to add those fields.

Data Relationships

Similar to SQL, GraphQL data can be related to other data. For example, in Onsight Flow you can:

- Get a list of users and the groups they belong to
- Get a list of groups and the users they contain

This is querying for the same data, but is instead being accessed from either direction.

3.1.1.1. More Information

There is a huge wealth of information on GraphQL and its features at graphql.org.

3.1.2. Trying out GraphQL

Within Onsight Flow, GraphQL is designed to be embedded into data clients (ie: software designed to extract data from Onsight Flow). But, in order to test out the queries, there are a number of tools that can be supported.

3.1.2.1. Onsight Flow Query Designer

Available within the [(Onsight Flow account required), the Query Designer can be used to test your GraphQL queries against the data in your Onsight Flow Team. Once you have tested out your query then you can export a GET or POST URL to import into your own system.

For more information on the query designer, see [Query Designer \(on page 43\)](#) .

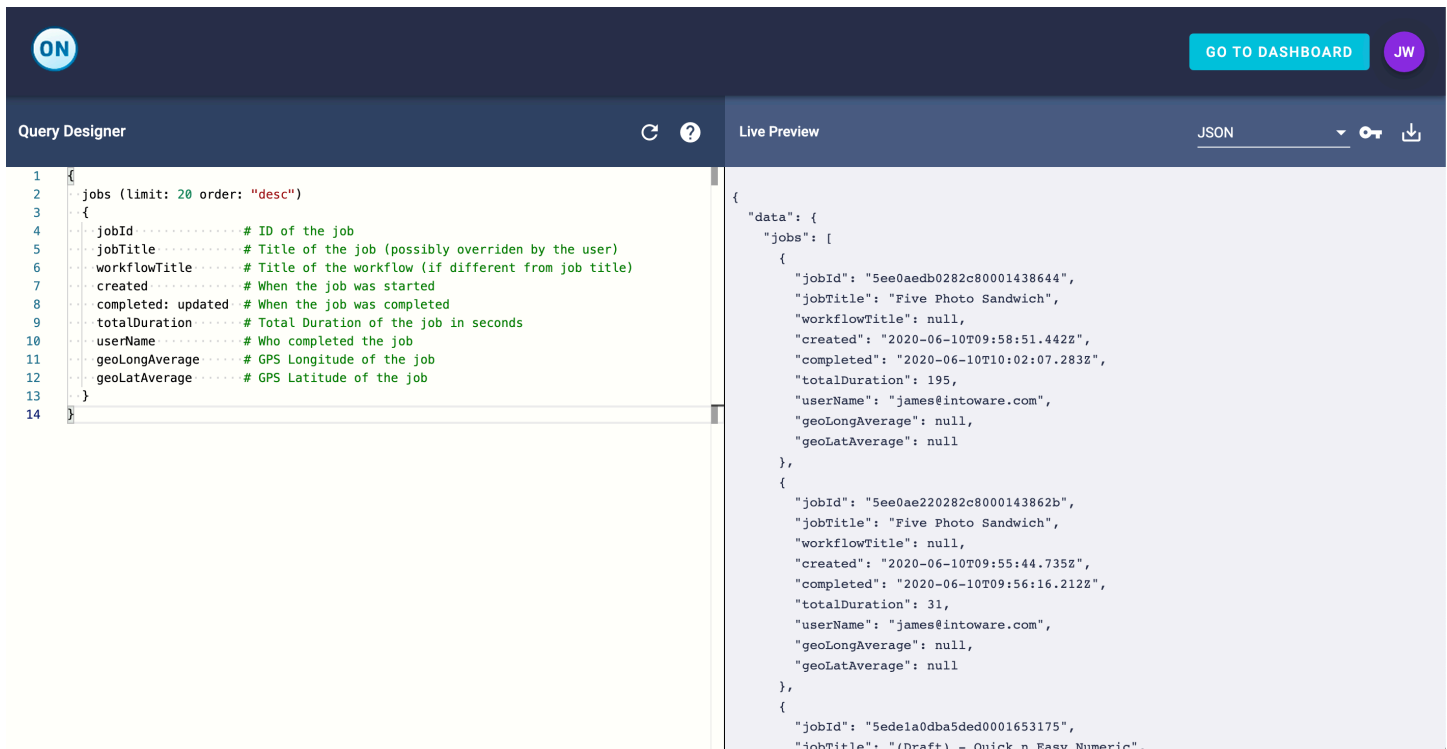


Figure 3-1 Onsignt Flow Query Designer

3.1.2.2. Postman

The excellent [Postman Tool](#) has built-in support for making GraphQL queries. To access the Onsignt Flow GraphQL endpoint, you'll need to fetch an [Accessing Onsignt Flow GraphQL API \(on page 40\)](#) in order to get access to the data.

Unlike the Onsignt Flow Query Designer, Postman supports GraphQL variables. This will allow you to design your queries for reuse so you can separate your template and variables. Onsignt Flow supports the full GraphQL specification for variables.

See the [GraphQL specification on variables](#) for more information

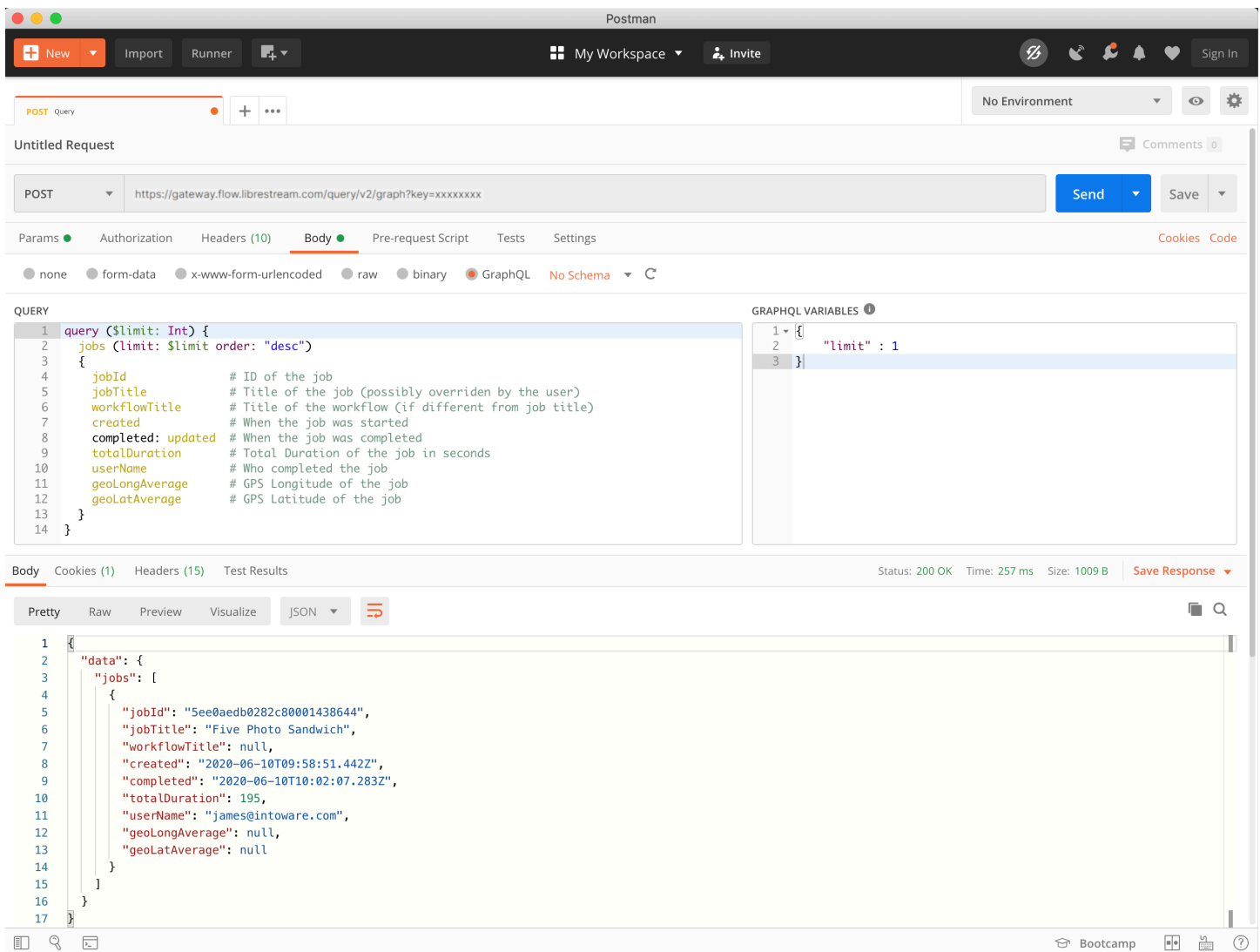


Figure 3-2 Postman Onsite Flow GraphQL Example

3.1.2.3. GET / POST Requests

If you'd like to build this functionality into your application, you can access the data via the direct HTTP GET or POST endpoints.

`https://gateway.flow.librestream.com/api/query/v2/graph`

Note, you will need to authenticate this request using one of the supported access key formats [Getting API Access \(on page 39\)](#)

For a more detailed explanation, see [GraphQL.org - Serving over HTTP](#).

GET Request

For a GET Request, simply take your GraphQL query and append it to the query endpoint as a URL-encoded string.

For example:

```
{
  jobs {
    jobId
    jobTitle
  }
}
```

becomes

<https://gateway.flow.librestream.com/api/query/v2/graph?query=%7B%20jobs%20%7B%20jobId%20jobTitle%20%7D%20%7D>

POST Request

For a POST Request, the same query endpoint is used, but the query should be encoded within a JSON payload.

```
{
  "query": "{\njobs\n{\njobId\njobTitle\n}\n}"
}
```

3.2. Getting Started

```
// Javascript code with syntax highlighting.
var fun = function lang(l) {
  dateformat.i18n = require('./lang/' + l)
  return true;
}
```

```
// GraphQL Code
query MyQuery {
  jobs {
    jobTitle
  }
}
```

After

3.3. Getting API Access

3.3.1. Accessing Onsight Flow GraphQL API

The Onsight Flow GraphQL endpoints can be accessed in the following ways:

- *Primary/Secondary Key* - A pre-generated set of keys, available within the Query Designer
- *Temporary Key* - A temporary key, sent as part of a HTTP Trigger
- *Access Tokens* - OAuth Access Tokens, available to authorized clients

In all cases, the access granted will allow you to query all completed job data for your team. See [GraphQL Overview \(on page 35\)](#) for an explanation of the types of queries that can be performed using this service.

3.3.2. Primary/Secondary Keys

2 pre-generated keys are created for each Onsight Flow team. These keys are available within the [Query Designer \(on page 43\)](#) (<https://dashboard.flow.librestream.com/query>) through the Keys dialog. From here you can regenerate the keys, and copy them to your clipboard for convenience.

Onsight Flow primarily employs OAuth2 as the security protocol for APIs however many of the tools that our users wish to use with their Onsight Flow data can't readily generate the bearer tokens for an OAuth2 API. Therefore the Query API employs access keys; two keys are available per team to use on all calls, these keys can be rotated and generated in line with an internal security protocol.

From a technical standpoint there is no difference between the Primary and Secondary keys and you can use them for your own purposes; typically though the Primary key is used on an ongoing basis and the Secondary key only used only temporarily whilst keys are rotated.

Follow this process to rotate your storage account keys:

1. Update the connection strings in your application(s) query(ies) to use the Secondary key.
2. Regenerate the Primary access key for your queries.
3. Update the connection strings in your application(s) query(ies) to use the new Primary key.
4. Regenerate the Secondary key in the same manner.

WARNING - If you regenerate a key, any existing links using that key will stop working

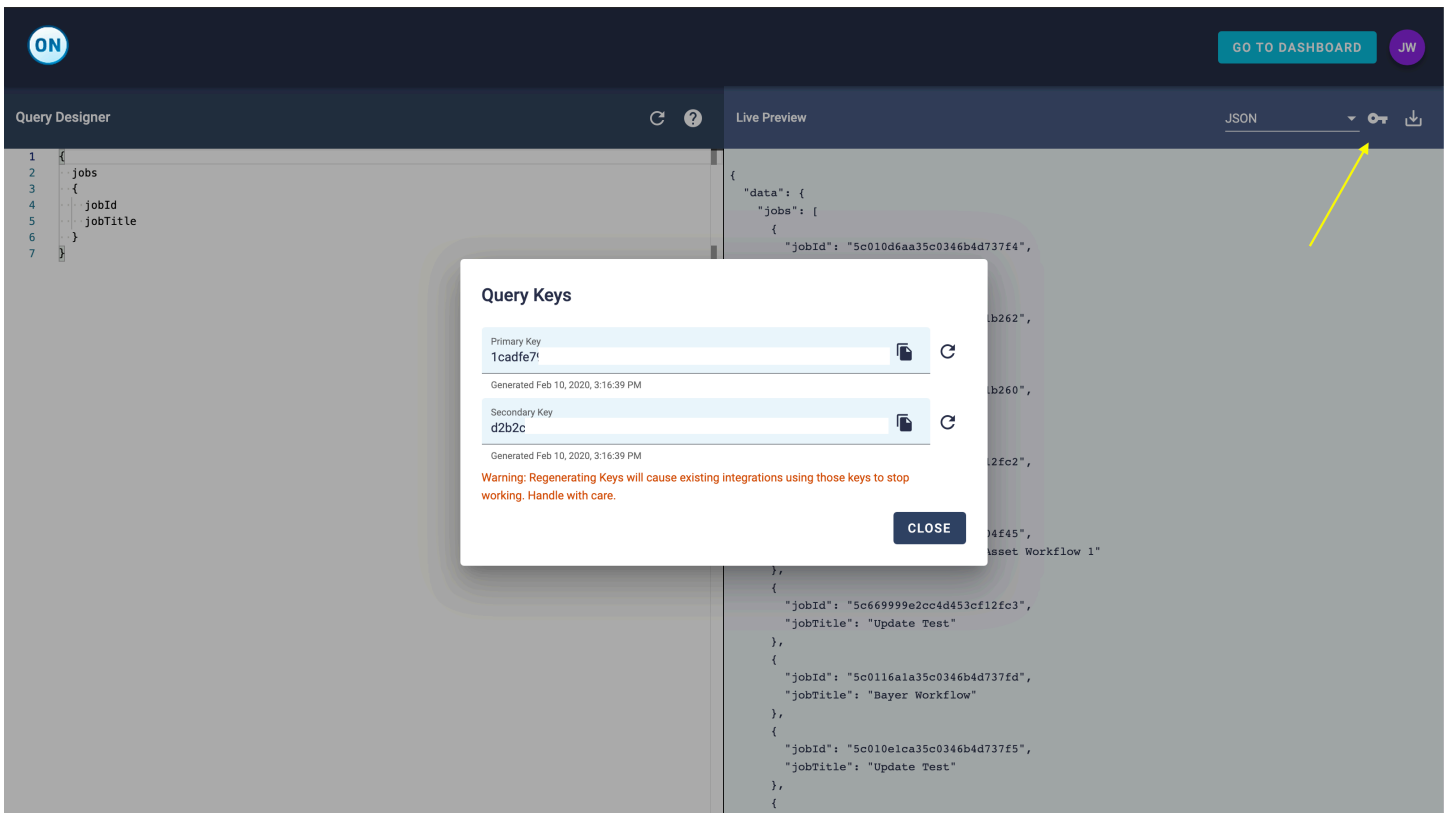


Figure 3-3 Primary/Secondary Key Dialog

3.3.2.1. Using Keys

To use a pre-generated key it should be appended to any Query URL using the parameter `key=MyCopiedKey`.

For example:

`https://gateway.flow.librestream.com/api/query/v2/graph?key=MyCopiedKey`

Note: When you export a query from the Query Designer, the generated URL will contain the Primary Key

3.3.3. Temporary Keys

If you have created a HTTP Trigger to receive Job Update/Complete data from Onsight Flow, the JSON Body will contain a temporary key. These keys function in the same way as the Primary/Secondary keys, but instead have a 1 hour lifecycle. Any system that responds to a HTTP Trigger can extract the key from the JSON payload and then use that key to make queries against the GraphQL Endpoint.

As with the Primary/Secondary Keys, the temporary key can be appended to the query string.

If your implementation requires access beyond 1 hour, consider using Access Tokens.

3.3.3.1. Example JSON Payload

```
{
  "data": {
    "jobs": [
      {
        ... Other Job Fields ...
        "key": {
          "value": "AAAAAAAABBBBBBCCCCCCCCCCCC",
          "expiry": "2020-06-11T14:03:07Z"
        }
      }
    ]
  }
}
```

3.3.4. Access Tokens

Access tokens can be acquired in a number of ways.

- A testing key can be generated using [Developer Tools](#)
- Via a custom API client

To get access to the Onsight Flow Client API, please [contact Support](#).

3.3.4.1. Generating Test Keys

From the [Onsight Flow Developer Tools](#) you can create a temporary access token with the **Query Read** permission. This will return an `access_token` and `id_token` that can be used in the Headers of a GraphQL request.

```
GET -
https://gateway.flow.librestream.com/api/query/v2/graph?query=%7B%20jobs%20%7B%20jobId%
20jobTitle%20%7D%20%7D

HEADERS:
Authorization : Bearer [PASTE ACCESS TOKEN HERE]
```

The testing Access Tokens comes with a 1hr lifetime and will return a `401 Unauthorized` error when expired. If you supply an Access Token with the wrong scope then the API will return `403 Forbidden`.

ON **GO TO DASHBOARD** JW

Policies ?

Select required policies and **Generate Access Token**

☐ Client_Admin	☐ Job_Modify	☐ Job_Read
☐ Job_ReadAll	☐ Query_Admin	☒ Query_Read
☐ Reports_Admin	☐ Reports_Create	☐ Reports_Modify
☐ Reports_Read	☐ Spoke_Client	☐ Team_Admin
☐ TeamSettings_Admin	☐ TeamSettings_Read	☐ Trigger_Admin
☐ Trigger_Execute	☐ Trigger_Read	☐ Workflow_Admin
☐ Workflow_Approve	☐ Workflow_Create	☐ Workflow_Delete
☐ Workflow_Execute	☐ Workflow_Modify	☐ Workflow_Read

GENERATE ACCESS TOKEN

Figure 3-4 Primary/Secondary Key Dialog

3.4. Query Designer

The OnSight Flow Query Designer is a developer tool to help design GraphQL queries against your OnSight Flow Team data.

You can access the query designer by selecting *Query* from the OnSight Flow Dashboard menu.

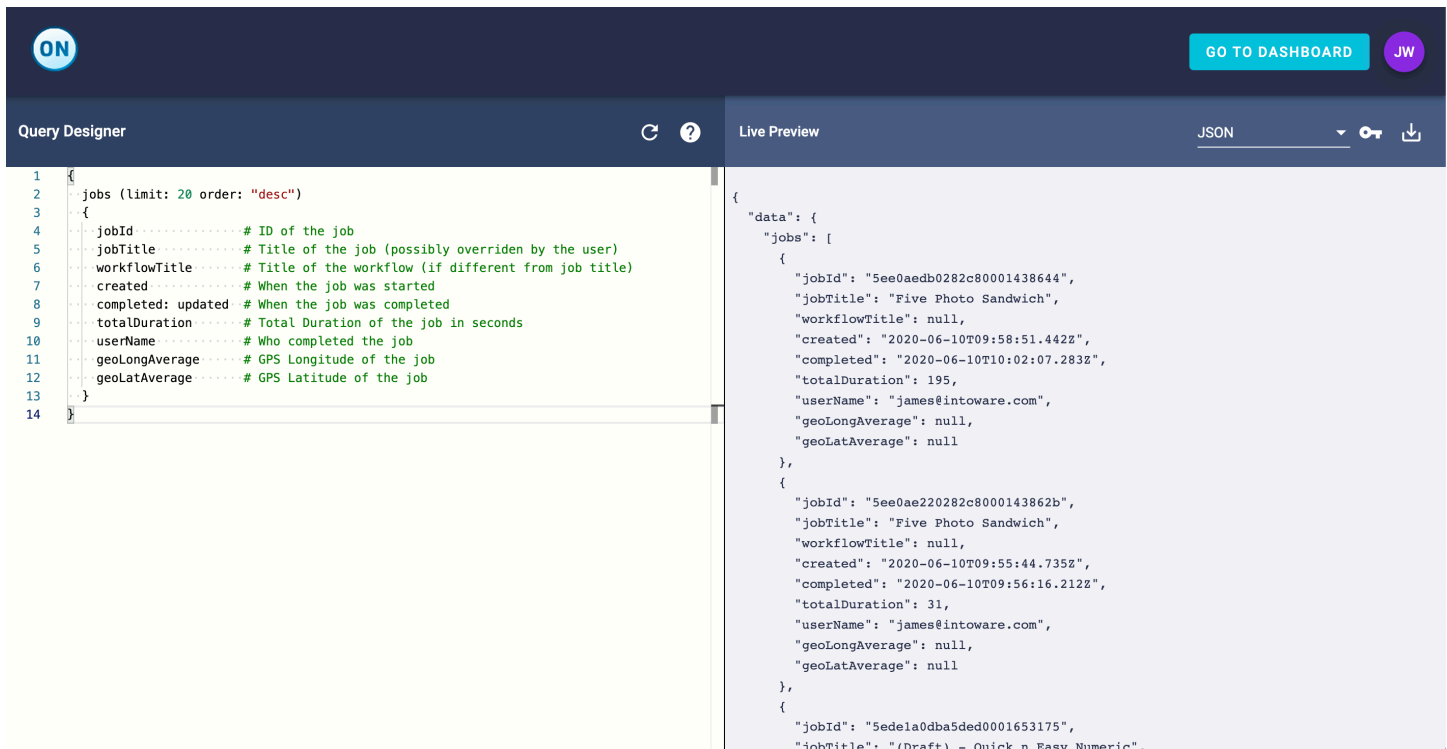


Figure 3-5 OnSight Flow Query Designer

Related information

[Writing Queries \(on page 44\)](#)

[Exporting Queries \(on page 44\)](#)

[CSV Queries \(on page 45\)](#)

3.4.1. Writing Queries

The Query Designer includes a [Monaco Editor](#) which provides basic syntax highlighting, auto-completion and other code editor features.

When you start typing your query, the text is automatically previewed and (if valid), a result will be returned in the preview window.

Any syntax errors will be displayed in the results.

See [Example Queries \(on page 63\)](#) for different use-cases.

3.4.2. Exporting Queries

When you are happy with your query, you can export the query to either:

- **HTTP GET** - A complete URL to paste directly into your browser
- **HTTP POST** - A URL and Body to be used in your code

The query will be exported either as JSON or CSV, depending on your preference in the Query Designer.

By default, the query will contain the Primary key, but you are free to change this to any on of the [Accessing OnSight Flow GraphQL API \(on page 40\)](#).

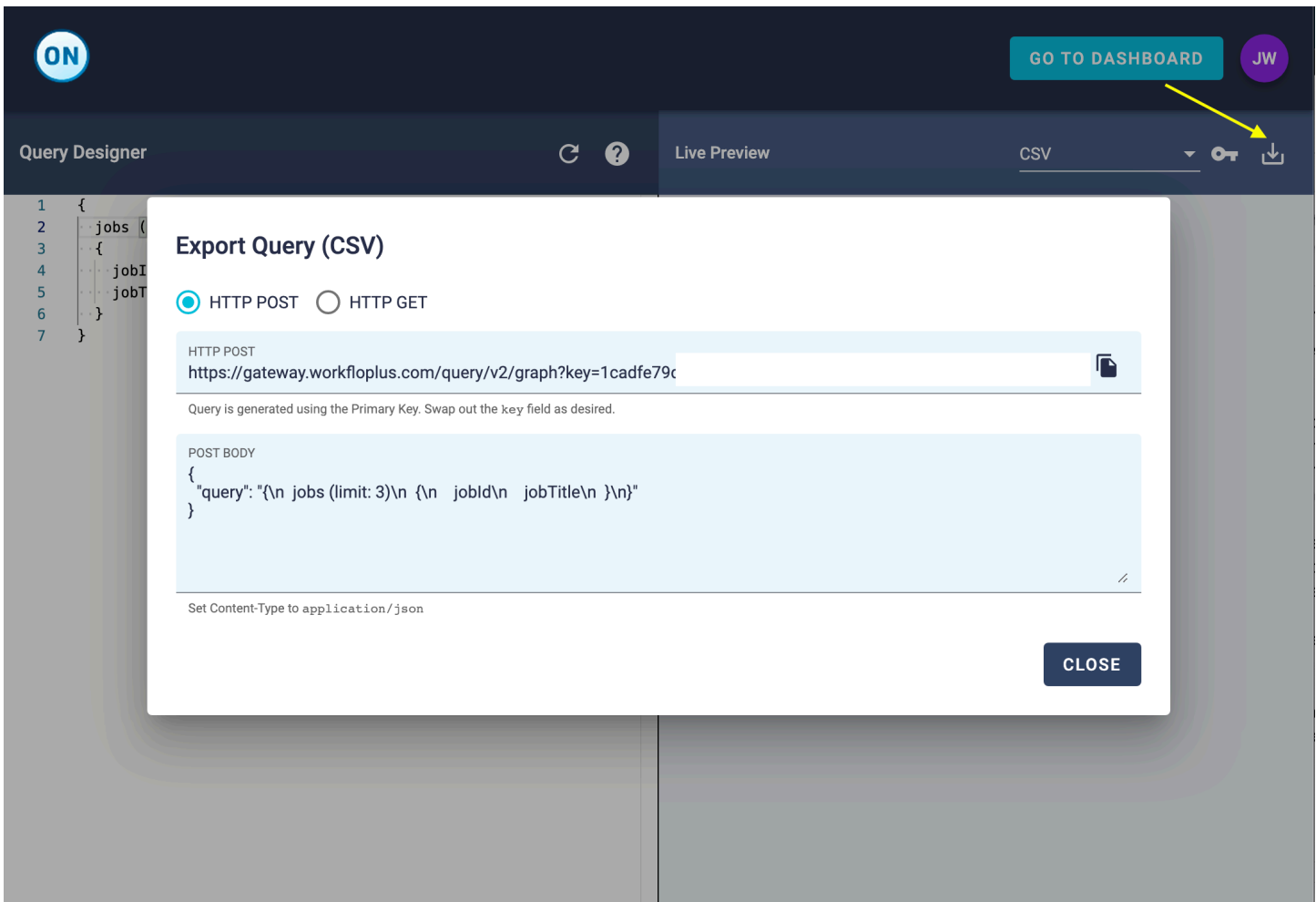


Figure 3-6 Query Designer Export

3.4.3. CSV Queries

By default, GraphQL renders any results to the JSON format. But, the OnSight Flow Query API can also render results to CSV.

Limitations:

- All errors are suppressed, only the `data` field is exported
- All arrays are flattened with a 0-based index in the column
- Each top-level item is converted into a CSV-row
 - Each sub-level item is flattened into a column

If your resulting data doesn't look right, switch back to JSON and check for any errors

It is not advised to produce a query that contains too many columns, or the resulting data may be too much for your integration.

To try out CSV, simply change the format toggle from JSON to CSV. You will see the results appear immediately.

To programmatically return CSV data from the GraphQL endpoints, append `&csv=true` to the query string. This works for both HTTP GET and HTTP POST requests.

Note: When queried in HTTP, the returned Content-Type will be `text/csv`.

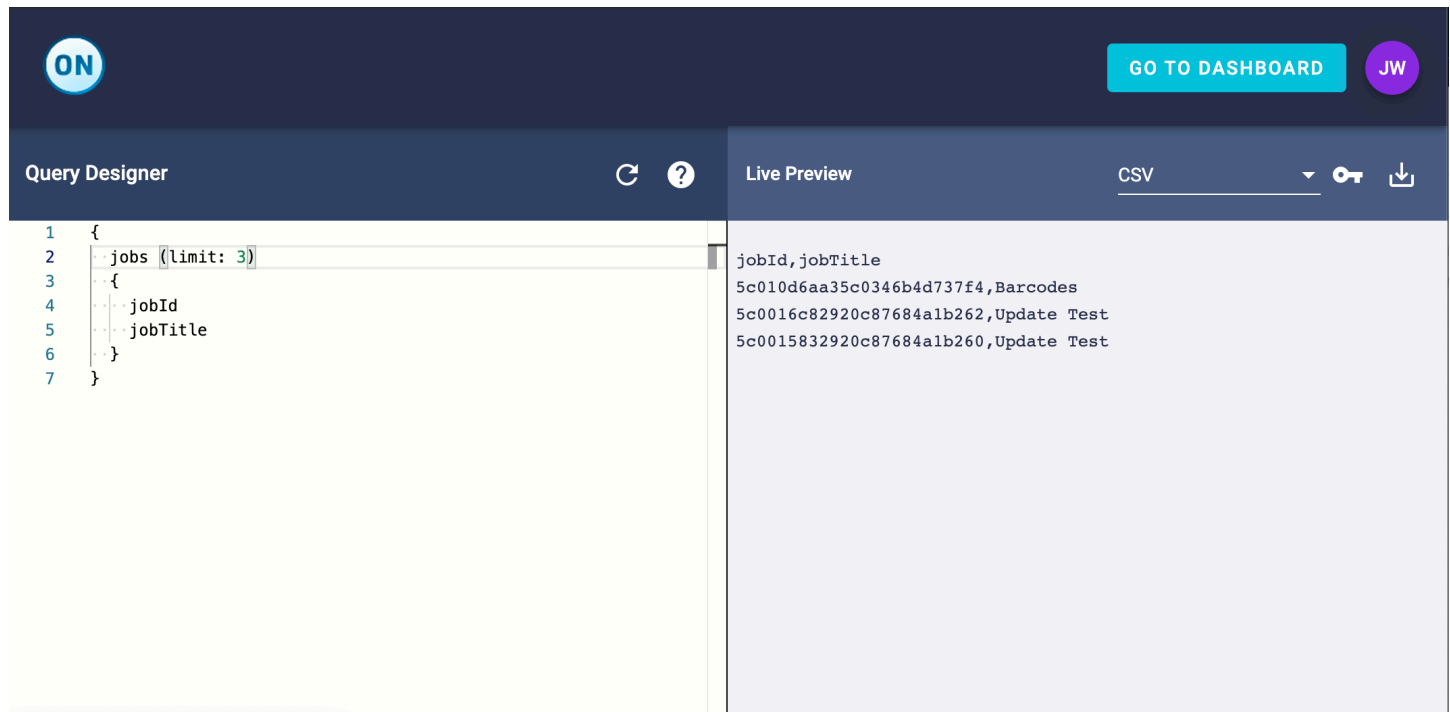


Figure 3-7 Query Designer CSV Output

3.5. Querying Jobs

Related information

[Introduction \(on page 46\)](#)

[Filtering on specific Jobs \(on page 47\)](#)

[Selecting Job Fields \(on page 52\)](#)

[Select Job Attachments \(on page 55\)](#)

3.5.1. Introduction

Querying for Jobs is the most common application of the OnSight Flow Query API

Related information

[Starting a Query \(on page 47\)](#)

3.5.1.1. Starting a Query

The base query structure for a job query is as follows (N.B. this query alone will not work as at least one field must be specified)

```
{
  jobs
  {
  }
}
```

3.5.2. Filtering on specific Jobs

By default the query will return the first 50 completed jobs for your team; however there are several query parameters that can be used to filter the query to return only the required jobs

Related information

[Filtering on Onsight Flow \(on page 47\)](#)

[Filtering on User\(s\) \(on page 48\)](#)

[Filtering on Job Identifier\(s\) \(on page 49\)](#)

[Filtering on Date and Time \(on page 50\)](#)

[Limit, Order & Pagination \(on page 51\)](#)

3.5.2.1. Filtering on Onsight Flow

There are several options for filtering on workflows but you can only apply one of them

Filtering on WorkflowId(s)

If you wish to match on one or more workflow Ids you can add this as part of the query

```
{
  jobs(workflowId: "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx")
  {
  }
}
```

```
{
  jobs(workflowId: ["xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
"yyyyyyyy-yyy-yyy-yyy-yyy-yyyyyyyyyyyy"])
  {
  }
}
```

Filtering on Workflow Title(s)

If you wish to match on one or more workflow titles you can add this as part of the query

```
{
  jobs(workflowTitle: "Inspection A")
  {
  }
}
```

```
{
  jobs(workflowTitle: ["Inspection A", "Inspection B"])
  {
  }
}
```

Filtering on Onsight Flow Tag(s)

If you wish to match on one or more workflow tags you can add this as part of the query; when matching on multiple workflows you can either match workflows that have at least one of the tags listed (workflowTagsOr) or alternatively only on workflows that have all of the tags listed (workflowTagsAnd)

```
{
  jobs(workflowTag: "Inspection")
  {
  }
}
```

```
{
  jobs(workflowTagsOr: ["Inspection", "Repair"])
  {
  }
}
```

```
{
  jobs(workflowTagsAnd: ["Inspection", "Site A", "Mechanical"])
  {
  }
}
```

3.5.2.2. Filtering on User(s)

There are two options for filtering on the user that completed a job but you can only apply one of them

Filtering on Username(s)

If you wish to match on one or more usernames you can add this as part of the query

```
{
  jobs(userName: "anne.worker@repairs.com")
  {
  }
}
```

```
{
  jobs(userName: ["inspector1@repairs.com", "inspector2@repairs.com"])
  {
  }
}
```

Filtering on UserId(s)

If you wish to match on one or more user Ids you can add this as part of the query

```
{
  jobs(userId: "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx")
  {
  }
}
```

```
{
  jobs(userId: ["xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx",
"yyyyyyyy-yyy-yyy-yyy-yyy-yyyyyyyyyyyyyy"])
  {
  }
}
```

3.5.2.3. Filtering on Job Identifier(s)

There are two options for filtering on specific job identifiers but you can only apply one of them

Filtering on Job Titles(s)

If you wish to match on one or more job titles you can add this as part of the query

```
{
  jobs(jobTitle: "Inspection 3A/5F")
  {
  }
}
```

```
{
  jobs(jobTitle: ["Inspection 3A/5F", "Inspection 3A/5G"])
  {
  }
}
```

Filtering on JobId(s)

If you wish to match on one or more job Ids you can add this as part of the query

```
{
  jobs(jobId: "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx")
  {
  }
}
```

```
{
  jobs(jobId: ["xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
"yyyyyyyy-yyy-yyy-yyy-yyy-yyyyyyyyyyyy"])
  {
  }
}
```

3.5.2.4. Filtering on Date and Time

There are options for filtering on the date and time that a job was both created and updated and also on, any combination of the following can be applied together

Filtering on jobs created since a given date and time

If you wish to filter on jobs created since a given date and time you can specify that as an ISO format datetime

```
{
  jobs(createdSince: "2019/06/01T08:00:00.000Z")
  {
  }
}
```

Filtering on jobs created before a given date and time

If you wish to filter on jobs created before a given date and time you can specify that as an ISO format datetime

```
{
  jobs(createdBefore: "2019/06/01T08:00:00.000Z")
  {
  }
}
```

Filtering on jobs updated since a given date and time

If you wish to filter on jobs updated since a given date and time you can specify that as an ISO format datetime

```
{
  jobs(updatedSince: "2019/06/01T08:00:00.000Z")
  {
  }
}
```

Filtering on jobs updated before a given date and time

If you wish to filter on jobs updated before a given date and time you can specify that as an ISO format datetime

```
{
  jobs(updatedBefore: "2019/06/01T08:00:00.000Z")
  {
  }
}
```

3.5.2.5. Limit, Order & Pagination

By default a query on the Jobs model will return data for up to the first 50 jobs that meet the criteria of the filter, however this number can be specified using the limit parameter, any value between 1 and 100 is allowed.

By default a query on the Jobs model will return data for the first n jobs however the order of the data can be controlled using the order parameter; this is equivalent to setting order = "asc", setting the order = "desc" will return the last n jobs.

As the number of jobs increases it becomes inefficient to pull down the entire data set everytime a data update is required, instead it is recommended to build your own store of the data. Pagination is the process of taking the data piece by piece to build up a full data set. We support a cursor based pagination solution, whereby everytime you query the data you can get the completedJobCursor for each job, cache the cursor for the last job in the list and then pass that back as a parameter next time you run the query to get the next n jobs.

```
{
  jobs(limit: 100 order: "desc" cursor: "5e3036f5e456fc00012e2b89")
  {
  }
}
```

3.5.3. Selecting Job Fields

Related information

[Selecting Fields on the Job Level \(on page 52\)](#)

[Selecting Fields on the Steps Level \(on page 52\)](#)

[Selecting Fields on the Substeps Level \(on page 53\)](#)

[Selecting Specific Steps \(on page 54\)](#)

3.5.3.1. Selecting Fields on the Job Level

There are a range of fields on the job level that can be included in the query and returned

```
{
  jobs
  {
    jobId
    completedJobCursor    # Cursor to be used for data pagination, relates to the
timestamp at which the job was uploaded
    teamName
    workflowId
    workflowVersionId
    workflowTitle
    workflowTags
    jobTitle
    created
    updated
    totalDuration          # Total Duration given in Seconds
    activeDuration         # Active Duration given in Seconds (equivalent to the
Total Duration minus any period where the Job was paused)
    status
    userId
    userName
    geoLongAverage         # Median average longitude across all completed steps
    geoLatAverage         # Median average latitude across all completed steps
    metadata               # Any metadata attached to the job displayed as an array
of name value pairs
    {
      name
      value
    }
  }
}
```

3.5.3.2. Selecting Fields on the Steps Level

Additionally the Steps field can be included on the Job level, this then returns an array of information for each step

```

{
  jobs
  {
    jobId
    steps
    {
      stepId
      uniqueStepId
      stepTitle
      stepDescription
      stepType
      started
      completed
      cancelled
      isCancelled
      isFormStep
      totalDuration
      activeDuration
      userId
      userName
      geoLat
      geoLong
      formLoopIteration
      note
      values
      value
    }
  }
}

```

3.5.3.3. Selecting Fields on the Substeps Level

Many users will not need to consider to substeps level and will be able to build all queries at the steps level. Substeps apply where Form Steps are used, here we have the concept of steps within steps and the substeps level allows you to break out the detail at each substep within a form.

```

{
  jobs
  {
    steps
    {
      substeps
      {
        stepId
        uniqueStepId
        stepTitle
        stepDescription
        stepType
        stepTag
        values
        value
      }
    }
  }
}

```

3.5.3.4. Selecting Specific Steps

Specific steps can be selected to be returned in the query and named at the same time

Step Search

A search term can be applied to the steps array, this will look for an exact match against each stepId, stepTag and then stepTitle. If none are found it will return null, if one or more matches are found it will return the first match that it finds.

The following query will return the jobId for each job, then the specified information for the first step that it finds that has a tag or title that matches "Instruction", then the specified information for the first step that it finds that has one of the stepIds in the list shown

```
{
  jobs
  {
    jobId
    InstructionStep:step(search: "Instruction")
    {
      stepId
      stepTitle
      totalDuration
    }
    RepairDetailsStep:step(search: ["a1cee52e-e950-4894-9f49-9e95862bb9a9",
"a1cee52e-e950-4894-9f49-9e95862bb9a1"])
    {
      stepId
      stepTitle
      totalDuration
    }
  }
}
```

Step Regex Search

Regex search works much like search except it can only match against a single term, however it can apply regex patterns for matching making it more powerful.

The following query will return the jobId for each job, then the specified information for the first step that it finds that includes the word "HSSE" anywhere in the step tag or title

Documentation on [regex patterns can be found here](#).

```
{
  jobs
  {
    jobId
    HsseStep:step(regexSearch: "HSSE")
    {
      stepId
      stepTitle
      totalDuration
    }
  }
}
```

3.5.4. Select Job Attachments

The attachments comprise all of the photos, images, videos, files etc that have been captured for a given job, there are different approaches available for constructing a query to extract your attachments

Related information

[Get All Attachments on a Job \(on page 55\)](#)

[Get Attachments on a Step or Substep \(on page 55\)](#)

[Attachment Model \(on page 56\)](#)

3.5.4.1. Get All Attachments on a Job

The most convenient way to simply extract details of all of the attachments on a job is to use the attachments field on the job

```
{
  jobs
  {
    jobId
    attachments
    {
      attachmentId
      # ...otherfields...
    }
  }
}
```

3.5.4.2. Get Attachments on a Step or Substep

Alternatively attachment details can be returned on the step or substep on which they were captured

```

{
  jobs
  {
    jobId
    steps
    {
      attachments
      {
        attachmentId
        # ...otherfields...
      }
    }
  }
}

```

```

{
  jobs
  {
    jobId
    steps
    {
      substeps
      {
        attachments
        {
          attachmentId
          # ...otherfields...
        }
      }
    }
  }
}

```

3.5.4.3. Attachment Model

The attachment model provides details of both the original attachment and additionally any thumbnails that Onsite Flow has generated; if you query a job shortly after it has been completed you may find that all attachments and thumbnails are not present in the result as they are still syncing or being generated

```

{
  jobs
  {
    jobId
    attachments
    {
      attachmentId
      stepId
      uniqueStepId
      stepTitle
      mimeType
      fileSize
      downloadUrl
      thumbnails
      {
        name
        created
        mimeType
        fileSize
        width
        height
        downloadUrl
      }
    }
  }
}

```

The downloadUrl for each attachment and each thumbnail is a temporary URL that can be used to download the file itself, for example if the response contains a downloadUrl

```

...
"fileSize": 14204,
"downloadUrl":
  "https://wfp.blob.core.windows.net/stepresources/myteam/63e39d97-d6cb-433b-b69c-2e2ed6c
76f37?sv=2019-02-02&sr=b&sig=HgS7dodYUEinIr3dXE%2BSh5isA3BBE8bZREg2I6m%2FvCo%3D&st=2020-
06-17T14%3A55%3A56Z&se=2020-06-17T16%3A00%3A56Z&sp=r",
...

```

That downloadUrl can be used in a Http request; the snippet below uses a downloadUrl returned by a query to save an attachment to a file called "myattachmentfile"

```

curl
  "https://wfp.blob.core.windows.net/stepresources/myteam/63e39d97-d6cb-433b-b69c-2e2ed6c
76f37?sv=2019-02-02&sr=b&sig=HgS7dodYUEinIr3dXE%2BSh5isA3BBE8bZREg2I6m%2FvCo%3D&st=2020-
06-17T14%3A55%3A56Z&se=2020-06-17T16%3A00%3A56Z&sp=r" --output myattachmentfile

```

If the downloadUrl expires a new one can be generated by re-querying the data

3.6. Ingest Data to Excel

The Onsight Flow GraphQL Query API supports integration with many tools; one of the more common requests is for users to extract their data from Onsight Flow into Excel. This example outlines an approach

for not only ingesting data into Excel but moreover for creating a live link between Excel and Onsight Flow so that the data set can be refreshed on demand from within Excel.



Figure 3-8 Onsight Flow > Excel

3.6.1. 1. Create the Query

In the following example selected data is extracted for each Inspection & Maintenance activity that is carried out. The query includes only jobs that are completed on the "Component Inspection & Maintenance" workflow.

For each job the query extracts selected information from the job and in addition it extracts defined fields on specified steps; for example on the *componentCodeStep* the value that is input is extracted as this is of interest, whereas on the *maintenanceStep* it is the step duration that is of interest and so this is specified in the query.

```

{
  jobs(workflowTitle: "Component Inspection & Maintenance" limit: 50 order: "desc")
  {
    jobId
    jobTitle
    workflowId
    workflowTitle
    userName
    created
    updated
    totalDuration
    activeDuration
    geoLongAverage
    geoLatAverage
    componentCodeStep: step(search: "componentcode")
    {
      value
    }
    conditionBeforeStep: step(search: "bd9a9c4a-0784-4f7c-9ccb-7c1f8739a3d9")
    {
      substeps
      {
        stepTitle
        value
      }
    }
    runningTemperatureBeforeStep: step(search:
"beforemaintenancerunningtemperature")
    {
      value
    }
    maintenanceStep: step(search: "ComponentMaintenance")
    {
      activeDuration
    }
    conditionAfterStep: step(search: "300d2f14-6a56-4ae4-a754-d017d8369376")
    {
      substeps
      {
        stepTitle
        value
      }
    }
    runUpStep: step(search: "Runup")
    {
      totalDuration
    }
    runningTemperatureAfterStep: step(search: "aftermaintenancerunningtemperature")
    {
      value
    }
  }
}

```

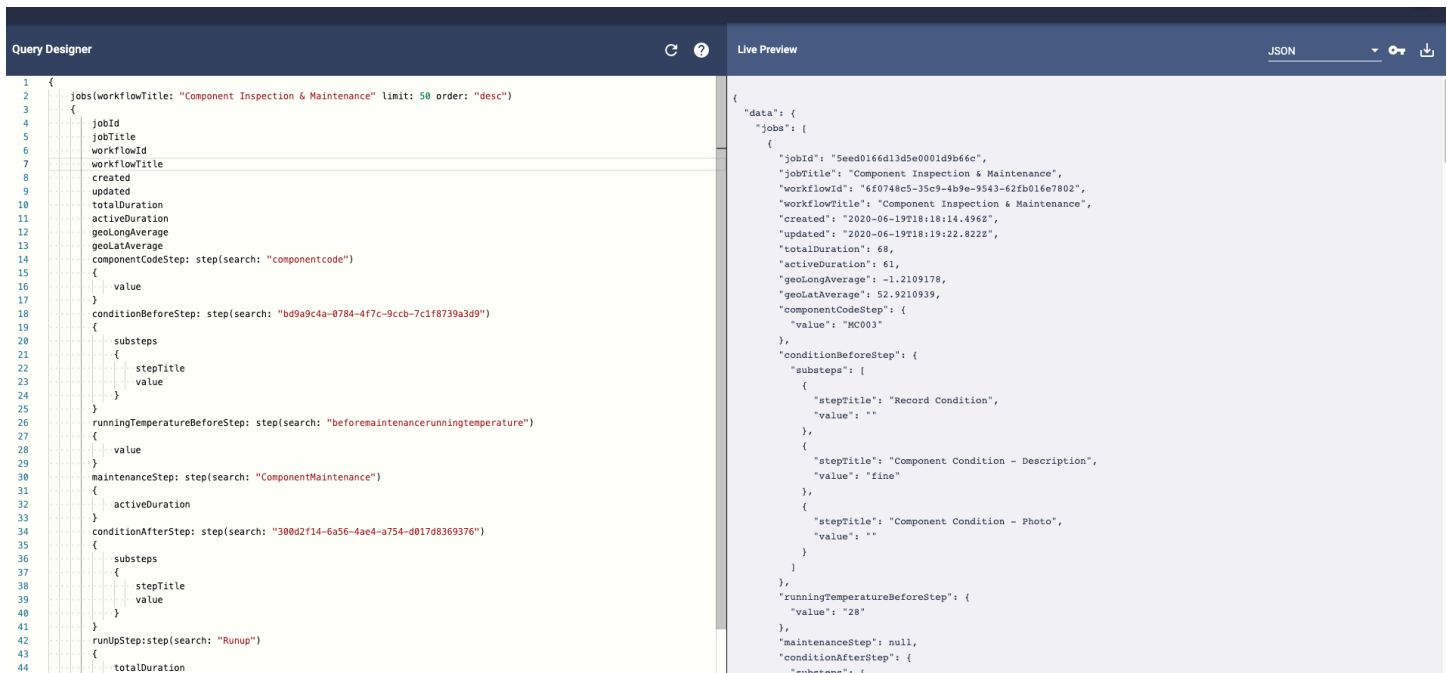


Figure 3-9 Create the Query in the Query Designer

3.6.2. 2. Select the Output Format

The defacto output format for a GraphQL query and for any HTTP request is JSON; Excel does support the conversion of JSON into tabular data however to make the process more convenient OnSight Flow also allows the user to extract the data in a flat format CSV, more information on setting the output format can be found [Query Designer \(on page 43\)](#)

3.6.3. 3. Execute Query from Excel

Microsoft's documentation on querying data from a web source can be found [here](#), the approach varies depending on the version of Excel. The example here uses version 16 of Excel.

Start a blank workbook, select the *Data* ribbon and then from within the Data ribbon select *From Web*. From the Query Designer [Exporting Queries \(on page 44\)](#), ensuring you use the GET request method along with the CSV format option and then paste the url into the dialog box that has opened in Excel.

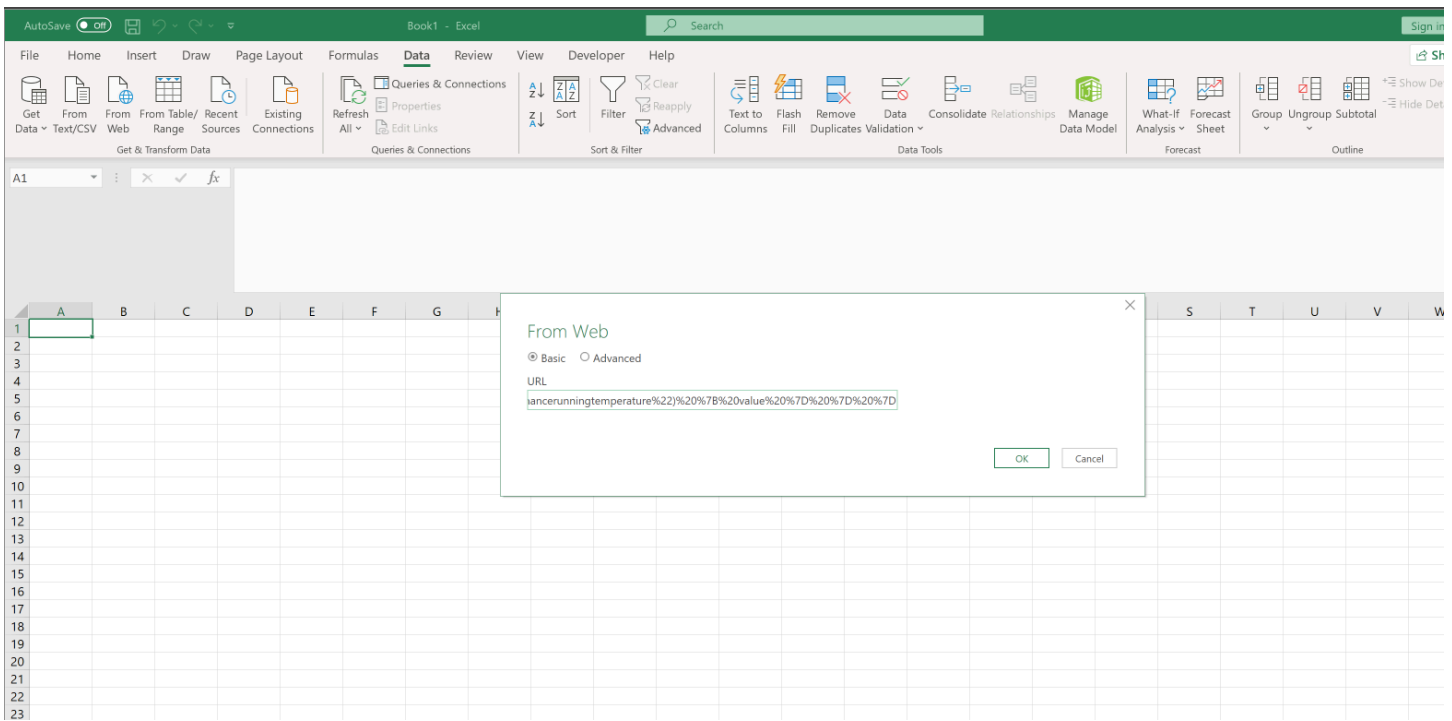


Figure 3-10 Paste in the copied url

After a short time Excel will display a preview of the table structure, if you are happy with the preview you can go ahead and click *Load*, alternatively you can click *Transform* to make refinements to the ingestion such as specifying that the first row should be treated as headers (Excel doesn't always infer this) or setting the data format of one or more columns before proceeding with the ingestion.

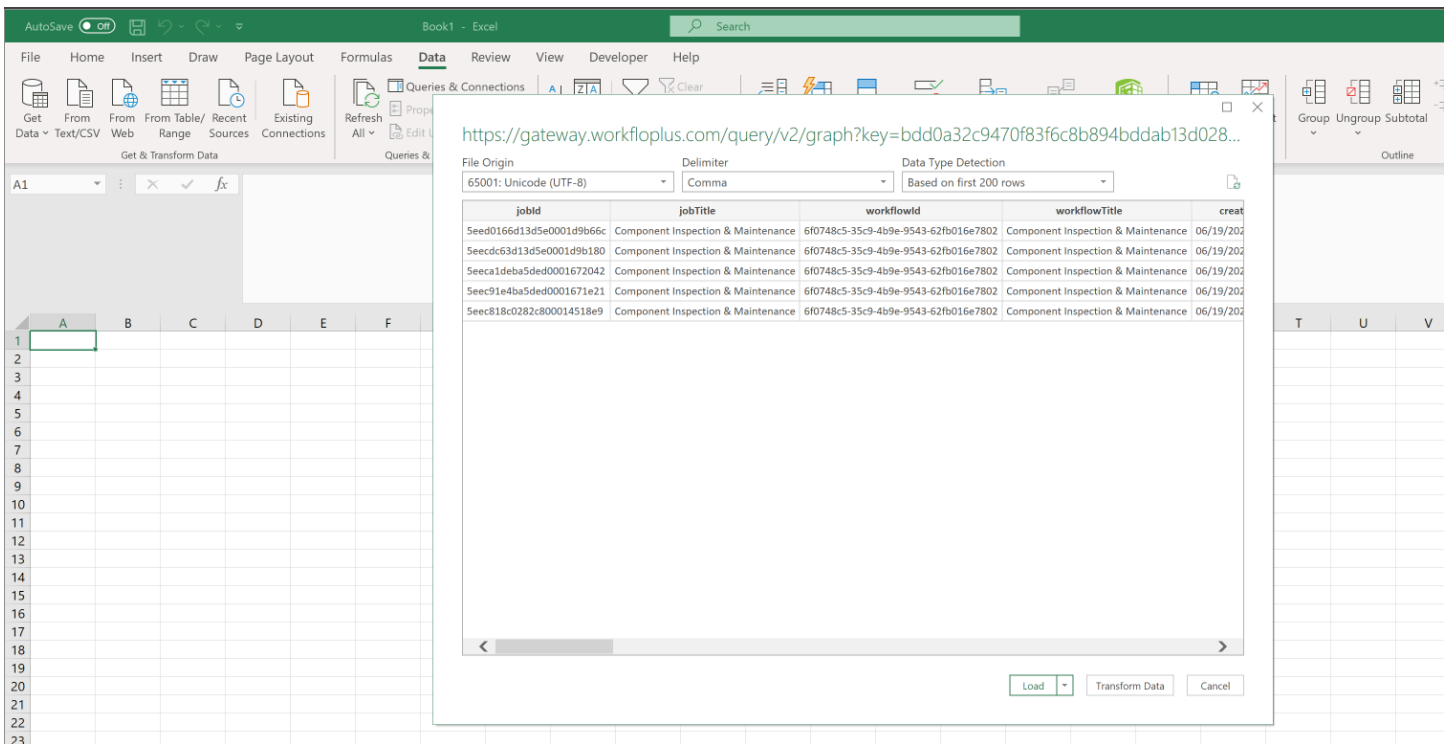


Figure 3-11 Preview the Data Format before proceeding

Once you have done this Excel will create a new sheet containing your Onsight Flow data.

	A	B	C	D	E	F
1	jobId	jobTitle	workflowId	workflowTitle	created	updated
2	5eed0166d13d5e0001d9b66c	Component Inspection & Maintenance	6f0748c5-35c9-4b9e-9543-62fb016e7802	Component Inspection & Maintenance	06/19/2020 18:18:14	06/19/2020 18:18:14
3	5eedc63d13d5e0001d9b180	Component Inspection & Maintenance	6f0748c5-35c9-4b9e-9543-62fb016e7802	Component Inspection & Maintenance	06/19/2020 15:40:18	06/19/2020 15:40:18
4	5eedc1deba5ded0001672042	Component Inspection & Maintenance	6f0748c5-35c9-4b9e-9543-62fb016e7802	Component Inspection & Maintenance	06/19/2020 11:30:37	06/19/2020 11:30:37
5	5eed91e4ba5ded0001671e21	Component Inspection & Maintenance	6f0748c5-35c9-4b9e-9543-62fb016e7802	Component Inspection & Maintenance	06/19/2020 10:22:28	06/19/2020 10:22:28
6	5eed818c0282c800014518e9	Component Inspection & Maintenance	6f0748c5-35c9-4b9e-9543-62fb016e7802	Component Inspection & Maintenance	06/19/2020 09:12:44	06/19/2020 09:12:44
7						
8						
9						
10						
11						
12						
13						
14						

Queries & Connections

Queries

Connections

1 query

graph?key=bdd0a32c9470f83f6c8b894bddab13d028e98e06c2b3b9...

5 rows loaded.

Figure 3-12 View the imported Data

3.6.4. 4. Manage the Query

3.6.4.1. Refresh

To refresh the data set to include the latest jobs click refresh within either the Data, Table Design or Query ribbon. *Refresh the data*

Book1 - Excel

Search

Formulas

Data

Review

View

Developer

Help

Table Design

Query

Queries & Connections

Refresh

Properties

Edit Links

Sort

Filter

Clear

Reapply

Advanced

Text to Columns

Flash Fill

Remove Duplicates

Data Validation

Consolidate

Relationships

Manage Data Model

Refresh All

Refresh

Refresh Status

Cancel Refresh

Connection Properties...

Figure 3-13 Excel - Refresh Data

3.6.4.2. Query Details

To view the details of the query click on the query in the Queries & Connections panel. *View Query details*

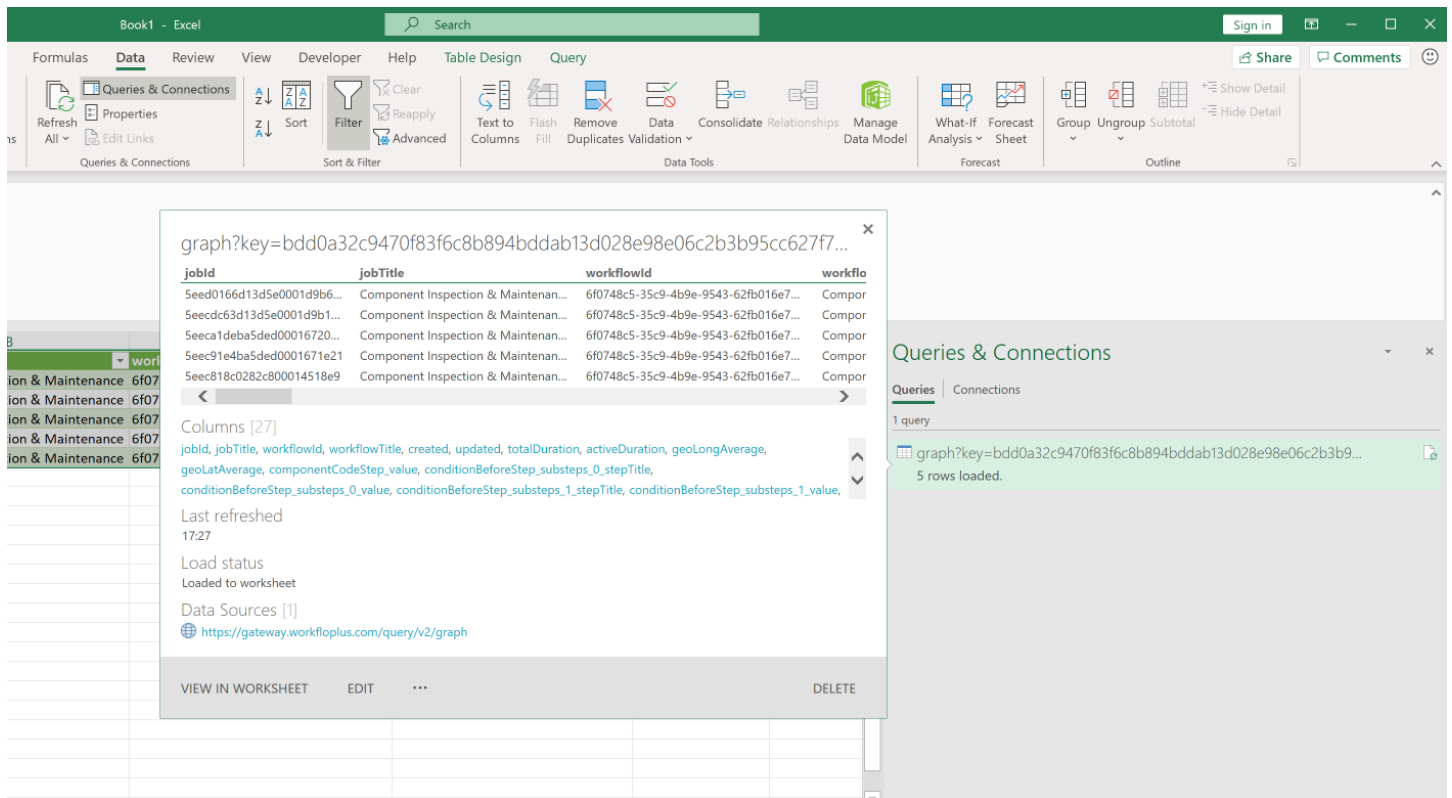


Figure 3-14 Excel - Query Details

3.6.5. A note on Power BI

The approach above leverages Microsoft's Power Query which is shared by both Excel and Power BI and therefore the method for ingesting data to Power BI is similar to that for Excel.

More to come on Power BI soon

3.7. Example Queries

This page contains a number of example queries that you can use in the Query Designer. All of these queries can be copied directly into the Query Designer as-is to provide a starting block for your own queries.

Related information

- [Get Latest Jobs \(on page 64\)](#)
- [Get All Reports \(on page 64\)](#)
- [Get All Attachments \(original file only\) \(on page 64\)](#)
- [Get All Attachment Thumbnails \(on page 65\)](#)
- [Get Substep Tags \(on page 65\)](#)
- [Get Job Level Data \(on page 66\)](#)
- [Get All Users \(on page 67\)](#)

3.7.1. Get Latest Jobs

Gets the latest 20 jobs with some basic metadata.

```
{
  jobs (limit: 20 order: "desc")
  {
    jobId
    jobTitle
    workflowTitle
    created
    completed
    totalDuration
    userName
    geoLongAverage
    geoLatAverage
  }
}
```

3.7.2. Get All Reports

Gets the latest 5 jobs containing with a list of generated PDF reports and their download link.

```
{
  jobs (limit: 5 order: "desc")
  {
    jobId
    jobTitle
    reports
    {
      generatedReportId
      generated
      templateName
      mimeType
      downloadUrl
    }
  }
}
```

3.7.3. Get All Attachments (original file only)

Gets the latest 5 jobs containing a list of all received attachments along with download links and metadata.

Note: See below for an example on Thumbnails

```
{
  jobs (limit: 5 order: "desc")
  {
    jobId
    jobTitle
    attachments
    {
      attachmentId
      stepId
      uniqueStepId
      stepTitle
      mimeType
      fileSize
      downloadUrl
    }
  }
}
```

3.7.4. Get All Attachment Thumbnails

For the latest 5 jobs, gets their attachments and any thumbnails generated. Usually there are 3 JPEG thumbnails (small, medium and large) generated for each image and video.

```
{
  jobs (limit: 5 order: "desc")
  {
    jobId
    jobTitle
    attachments
    {
      attachmentId
      thumbnails
      {
        name
        created
        mimeType
        fileSize
        width
        height
        downloadUrl
      }
    }
  }
}
```

3.7.5. Get Substep Tags

For the latest 5 jobs, get a list of all step tags captured during execution.

```

{
  jobs (limit: 5 order: "desc")
  {
    jobId
    jobTitle
    steps
    {
      substeps
      {
        stepTag
      }
    }
  }
}

```

3.7.6. Get Job Level Data

For the latest 5 jobs, gather more detailed metadata on the job.

```

{
  jobs (limit: 5 order: "desc")
  {
    jobId
    completedJobCursor
    teamName
    workflowId
    workflowVersionId
    workflowTitle
    workflowTags
    jobTitle
    created
    updated
    totalDuration
    activeDuration
    status
    userId
    userName
    geoLongAverage
    geoLatAverage
    metadata
    {
      name
      value
    }
  }
}

```

3.7.7. Get All Users

Gets the list of all users and their groups

```
{
  users {
    emailAddress
    name
    userGroups {
      name
    }
  }
}
```


4. Collections

Collections can be managed via the APIs

Related information

[Get Collection \(on page 69\)](#)

[Get Collections \(on page 70\)](#)

[Update Collection Name \(on page 71\)](#)

[Create Collection \(on page 72\)](#)

[Delete Collection \(on page 73\)](#)

[Upload Collection \(on page 73\)](#)

4.1. Get Collection

4.1.1. HTTP Request

```
GET https://gateway.flow.librestream.com/collections/v1/collection/{collection_id}
```

4.1.2. Authorization

Bearer Token

Policy Any Valid Token

4.1.3. Headers

Key	Value
Authorization	Bearer {token}

4.1.4. Path Parameters

Parameter	Type
collection_id	string

4.1.5. Response Body

4.1.5.1. JSON Representation

```
{
  "collections":[
    {
      "collectionId":"5e8c434572v84700018ffb2c",
      "userId":"5d8cacd37811130001e71db0",
      "userName":"colin.pope@solarenergy.com",
      "created":"2020-04-07T09:09:25.082Z",
      "lastUpdated":"2020-04-07T09:09:25.082Z",
      "teamName":"collectionteam",
      "name":"Collection 1",
      "dataLastChanged":"2020-04-07T14:08:21.727Z",
      "headerNames":[
        "PenID",
        "AnimalID",
        "AnimalName",
        "InspectionTime"],
      "recordCount":0,
      "csvFileName":"5e8c895578c444000190048e",
      "csvFileSize":693,
      "csvFileNameHistory":{
        "5e8c434f78c84700018ffb30":"2020-04-07T09:09:35.563Z",
        "5e8c895578c847000190048e":"2020-04-07T14:08:21.727Z"
      },
      "csvDownloadUrl":"https://
wfp27dev.blob.core.windows.net/collectionuploads/collectionteam/5e8c895578c444000190048e
?sv=2019-02-02\u0026sr=b\u0026sig=RdqRRlqBxHzuTYBpYQhZyQ0wbBsPxGlWCa8PqnWH7EE%3D\u0026st
=2021-09-14T05%3A41%3A53Z\u0026se=2021-09-14T06%3A46%3A53Z\u0026sp=r"
    }
  ]
}
```

4.2. Get Collections

4.2.1. HTTP Request

```
GET https://gateway.flow.librestream.com/collections/v1/collection
```

4.2.2. Authorization

Bearer Token

Policy Any Valid Token

4.2.3. Headers

Key	Value
Authorization	Bearer {token}

4.2.4. Response Body

4.2.4.1. JSON Representation

```
{
  "collections": [
    {
      "collectionId": "5e8c434572v84700018ffb2c",
      "userId": "5d8cacd37811130001e71db0",
      "userName": "colin.pope@solarenergy.com",
      "created": "2020-04-07T09:09:25.082Z",
      "lastUpdated": "2020-04-07T09:09:25.082Z",
      "teamName": "collectionteam",
      "name": "Collection 1",
      "dataLastChanged": "2020-04-07T14:08:21.727Z",
      "headerNames": [
        "PenID",
        "AnimalID",
        "AnimalName",
        "InspectionTime"],
      "recordCount": 0,
      "csvFileName": "5e8c895578c444000190048e",
      "csvFileSize": 693,
      "csvFileNameHistory": {
        "5e8c434f78c84700018ffb30": "2020-04-07T09:09:35.563Z",
        "5e8c895578c847000190048e": "2020-04-07T14:08:21.727Z"
      },
      "csvDownloadUrl": "https://wfp27dev.blob.core.windows.net/collectionuploads/collectionteam/5e8c895578c444000190048e?sv=2019-02-02\u0026sr=b\u0026sig=RdqRRlqBxHzuTYBpYQhZyQ0wbBsPxGLwCa8PqnWH7EE%3D\u0026st=2021-09-14T05%3A41%3A53Z\u0026se=2021-09-14T06%3A46%3A53Z\u0026sp=r"
    }
  ]
}
```

4.3. Update Collection Name

4.3.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/collections/v1/collection/{collection_id}
```

4.3.2. Authorization

Bearer Token

Policy Any Valid Token

4.3.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Type	application/json

4.3.4. Path Parameters

Parameter	Type
collection_id	string

4.3.5. Request Body

4.3.5.1. JSON Representation

```
{
  "name": "New Collection Name"
}
```

4.4. Create Collection

4.4.1. HTTP Request

```
POST https://gateway.flow.librestream.com/collections/v1/collection
```

4.4.2. Authorization

Bearer Token

Policy Any Valid Token

4.4.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Type	application/json

4.4.4. Request Body

4.4.4.1. JSON Representation

```
{
  "name": "New Collection"
}
```

4.4.5. Response Body

4.4.5.1. JSON Representation

```
{
  "collectionId": "61403c1230e5a30001d4440b"
}
```

4.5. Delete Collection

4.5.1. HTTP Request

```
DEL https://gateway.flow.librestream.com/collections/v1/collection/{collection_id}
```

4.5.2. Authorization

Bearer Token

Policy Any Valid Token

4.5.3. Headers

Key	Value
Authorization	Bearer {token}

4.5.4. Path Parameters

Parameter	Type
collection_id	string

4.6. Upload Collection

4.6.1. Overview

The Collection Upload API provides a way to reliably upload files of upto 50mb by blocking them into a sequence of parts upto 100kb that can be uploaded individually.

By using this API the application uploads a file in part, allowing it to recover from a failed request more reliably. It means an application only needs to retry the upload of a single part instead of the entire file.

An additional benefit of blocked uploads is that parts can be uploaded in parallel, allowing for a potential performance improvement.

The process for uploading a collection file requires a couple of API calls to be made

1. [Create Collection Upload Session \(on page 74\)](#): This session is used as a reference for the later calls and sets expectations for the block sizes and files sizes.
2. [File is Uploaded \(on page 75\)](#): The file is uploaded as a series of blocks (N.B. if the file is small enough it can be uploaded as a single block).

4.6.2. Notes

- Collection file size limit is 50 MB
- File block size limit is 100 KB

Related information

[Create Collection Upload Session \(on page 74\)](#)

[Upload Collection File \(on page 75\)](#)

4.6.3. Create Collection Upload Session

4.6.3.1. HTTP Request

```
POST https://gateway.flow.librestream.com/collections/v1/upload
```

4.6.3.2. Authorization

Bearer Token

Policy [Workflow_Create \(on page 32\)](#)

4.6.3.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Type	application/json

4.6.3.4. Request Body

4.6.3.4.1. JSON Representation

```
{
  "blockSize":102400,
  "fileSize":3134,
  "friendlyFileName":"ExampleCSV.csv",
  "collectionId":"61403c1230e5a30001d4440b"
}
```

4.6.3.5. Response Body

4.6.3.5.1. JSON Representation

```
{
  "id":"61404bfc60e5c32001d44625"
}
```

4.6.4. Upload Collection File

4.6.4.1. HTTP Request

POST https://gateway.flow.librestream.com/collections/v1/upload/{upload_id}

4.6.4.2. Authorization

Bearer Token

Policy [Workflow_Create](#) (on page 32)

4.6.4.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Type	text/csv
Content-Length	{blockSize}
Content-Range	bytes {i x blockSize}-{(i+1) x blockSize}-1/{fileSize}

4.6.4.4. Path Parameters

Parameter	Type
upload_id	string

4.6.4.5. Request Body

```
{block_bytes}
```

4.6.4.6. Response Body

4.6.4.6.1. If the block was the final block and all blocks been uploaded a 204 is returned

4.6.4.6.2. If the block was not the final block but has been uploaded a 202 is returned

4.6.4.7. Notes

I have a file of size 150 KB which I will upload in blocks of 100 KB, I multiply both of these by 1024 to calculate the file size and block size

```
BLOCKSIZE = 102400  
FILESIZE = 153600
```

5. Groups

Groups can be managed via the APIs

Related information

[Get All Groups \(on page 77\)](#)

[Create Group \(on page 78\)](#)

[Get Group Details \(on page 78\)](#)

[Update Group Name \(on page 79\)](#)

[Update Group Permissions \(on page 80\)](#)

[Add Users to Group \(on page 81\)](#)

[Remove Users from Group \(on page 82\)](#)

[Delete Group \(on page 83\)](#)

5.1. Get All Groups

5.1.1. HTTP Request

```
GET https://accounts.flow.librestream.com/api/group
```

5.1.2. Authorization

Bearer Token

Policy Any Valid Token

5.1.3. Headers

	Key	Value
	Authorization	Bearer {token}

5.1.4. Response Body

5.1.4.1. JSON Representation

```
[
  {
    "id": "5e4a95c36abe6e0001f80592",
    "name": "Example Group 1"
  },
  {
    "id": "5f367fe76147f40001def972",
    "name": "Example Group 2"
  }
]
```

5.2. Create Group

5.2.1. HTTP Request

```
POST https://gateway.flow.librestream.com/group
```

5.2.2. Authorization

Bearer Token

Policy [Team_Admin](#) (on page 30)

5.2.3. Headers

Key	Value
Authorization	Bearer {token}

5.2.4. Request Body

5.2.4.1. JSON Representation

```
{
  "name": "Group Name"
}
```

5.2.5. Response Body

5.2.5.1. JSON Representation

```
{
  "id": "5e4a95c36abe6e0001f80592",
  "name": "Group Name"
}
```

5.3. Get Group Details

5.3.1. HTTP Request

```
GET https://gateway.flow.librestream.com/group/{group_id}
```

5.3.2. Authorization

Bearer Token

Policy Any Valid Token

5.3.3. Headers

Key	Value
Authorization	Bearer {token}

5.3.4. Path Parameters

Parameter	Type
group_id	string

5.3.5. Response Body

5.3.5.1. JSON Representation

```
{
  "groupId": "5e4a95c36abe6e0001f80592",
  "teamName": "Example Team",
  "name": "Example Group 1",
  "lowercaseName": "example group 1",
  "created": "2020-03-02T11:08:30Z",
  "users": [
    "5d8cab89860b81efr16b196a",
    "5d91f997ccfe6acf01dc9dc1"
  ],
  "permissions": {
    "dataViewer": true,
    "editor": true,
    "modifier": true
  }
}
```

5.4. Update Group Name

5.4.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/api/group/{group_id}
```

5.4.2. Authorization

Bearer Token

Policy Any Valid Token

5.4.3. Headers

Key	Value
Authorization	Bearer {token}

5.4.4. Path Parameters

Parameter	Type
group_id	string

5.4.5. Request Body

5.4.5.1. JSON Representation

```
{  
  "name": "New Group Name"  
}
```

5.4.6. Response Body

5.4.6.1. JSON Representation

```
{  
  "id": "5e4a95c36abe6e0001f80592",  
  "name": "New Group Name"  
}
```

5.5. Update Group Permissions

5.5.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/api/group/{group_id}/permissions
```

5.5.2. Authorization

Bearer Token

Policy [Team_Admin \(on page 30\)](#)

5.5.3. Headers

Key	Value
Authorization	Bearer {token}

5.5.4. Path Parameters

Parameter	Type
group_id	string

5.5.5. Request Body

5.5.5.1. JSON Representation

```
{
  "dataViewer": "true",
  "editor": "true",
  "modifier": "true",
  "executor": "true",
  "approver": "false"
}
```

5.6. Add Users to Group

5.6.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/api/group/{group_id}/users
```

5.6.2. Authorization

Bearer Token

Policy [Team_Admin \(on page 30\)](#)

5.6.3. Headers

Key	Value
Authorization	Bearer {token}

5.6.4. Path Parameters

Parameter	Type
group_id	string

5.6.5. Request Body

5.6.5.1. JSON Representation

```
[  
  "5dd501931da0b45001bf0da3",  
  "5dd501931da0b30551bf0ra3"  
]
```

User Ids of the users to be added are given in json body.

5.7. Remove Users from Group

5.7.1. HTTP Request

```
DEL https://gateway.flow.librestream.com/api/group/{group_id}/users
```

5.7.2. Authorization

Bearer Token

Policy [Team_Admin \(on page 30\)](#)

5.7.3. Headers

Key	Value
Authorization	Bearer {token}

5.7.4. Path Parameters

Parameter	Type
group_id	string

5.7.5. Request Body

5.7.5.1. JSON Representation

```
[  
  "5dd501931da0b45001bf0da3",  
  "5dd501931da0b30551bf0ra3"  
]
```

User Ids of the users to be added are given in json body.

5.8. Delete Group

5.8.1. HTTP Request

```
DEL https://accounts.flow.librestream.com/api/group/{group_id}
```

5.8.2. Authorization

Bearer Token

Policy [Team_Admin \(on page 30\)](#)

5.8.3. Headers

Key	Value
Authorization	Bearer {token}

5.8.4. Path Parameters

Parameter	Type
group_id	string

6. Jobs

Jobs can be managed via the APIs

Related information

[Get All Jobs \(on page 85\)](#)

[Get Jobs by Workflow Id \(on page 87\)](#)

[Get Snapshots of a Job \(on page 89\)](#)

[Get Job by Client Job Id \(on page 90\)](#)

[Get Job by Job Id \(on page 92\)](#)

[Get My Jobs \(on page 94\)](#)

[Abort a Job \(on page 96\)](#)

[Delete a Job \(on page 97\)](#)

[Assign User to a Job \(on page 97\)](#)

[Exclude a Job \(on page 98\)](#)

[Add Metadata to a Job \(on page 99\)](#)

[Create a Job \(on page 99\)](#)

[Abort Jobs for an User \(on page 102\)](#)

[Basic Sync \(on page 103\)](#)

[Update a Job \(on page 105\)](#)

6.1. Get All Jobs

6.1.1. HTTP Request

```
GET https://gateway.flow.librestream.com/jobs/v1/?{skip}&{limit}
```

6.1.2. Authorization

Bearer Token

Policy [Job_ReadAll \(on page 26\)](#)

6.1.3. Headers

Key	Value
Authorization	Bearer {token}

6.1.4. Query Parameters

Parameter	Type
skip	integer
limit	integer

NB: Skip - To skip first {skip} number of jobs , Limit - To limit the results to {limit} number of jobs. Default limit is 20.

6.1.5. Response Body

6.1.5.1. JSON Representation

```
[
  {
    {
      "jobId": "5fa2a458980724000149rq40",
      "metadata": {
        "clientJobId": "8a07b1c0-56ba-433d-8a24-adddgb3e150c",
        "workflowId": "a454a699-173f-429e-b6cd-fe9d92d0a1er",
        "workflowVersionId": "c683d13a-4cda-4683-1276-e7fcfb9c1c52",
        "jobTitle": "Example Workflow Job 1",
        "metadata": {
          "ScheduledJobId": "5fa2a3d17afcb9000179ad01",
          "ScheduledTaskId": "c95a93e0-b24f-44c3-a62f-7304c7ea7f2a"
        }
      },
      "created": "2020-11-04T12:53:42.266Z",
      "updated": "2020-11-04T12:53:46.676Z",
      "modified": "2020-11-04T12:53:47Z",
      "status": "Paused",
      "userId": "5d8cac54860b8100016b1987",
      "currentStep": {
        "singleStep": {
          "values": [
            ],
          "valueResourceIds": [
            ],
          "uniqueStepId": "0ffad266-89e0-4659-936b-c0f92d043c81",
          "previousUniqueStepId": "",
          "userId": "5d8cac54860b8103316b1987",
          "userName": "api.user1@example.com",
          "deviceId": null,
          "stepId": "41443154-f511-478b-bcca-849440421061",
          "stepNumber": 1,
          "stepTitle": "Enter Text",
          "stepDescription": "",
          "stepType": "Text",
          "connectionType": null,
          "started": "2020-11-04T12:53:42.687Z",
          "completed": null,
          "updated": "2020-11-04T12:53:46.664Z",
          "cancelled": null,
          "timeEvents": {
            "2020-11-04T12:53:46.664Z": 0
          }
        },
        "note": null,
        "noteResourceFileNames": null,
        "coordinates": null,
        "parentStepId": null,
        "parentUniqueStepId": null,
        "parentStepTitle": null,
        "parentStepDescription": null,
      }
    }
  }
]
```

```

        "metadata":{
            "tags":""
        },
        "reportStep":null
    },
    "abandonReason":null,
    "username":"api.user1@example.com",
    "workflowName":"Example Workflow"
},
"completedSteps":null,
"cancelledSteps":null,
"team":"apiteam",
"hierarchyStack":null,
"packetIds":null,
"jobExclusion":[
],
"excluded":false
}
]

```

6.2. Get Jobs by Workflow Id

6.2.1. HTTP Request

GET https://gateway.flow.librestream.com/jobs/v1/workflow/{workflow_id}

6.2.2. Authorization

Bearer Token

Policy [Job_ReadAll](#) (on page 26)

6.2.3. Headers

Key	Value
Authorization	Bearer {token}

6.2.4. Path Parameters

Parameter	Type
workflow_id	string

6.2.5. Response Body

6.2.5.1. JSON Representation

```
[
  {
    "jobId": "5fa2a458980724000149rq40",
    "metadata": {
      "clientJobId": "8a07b1c0-56ba-433d-8a24-adddgb3e150c",
      "workflowId": "a454a699-173f-429e-b6cd-fe9d92d0a1er",
      "workflowVersionId": "c683d13a-4cda-4683-1276-e7fcfb9c1c52",
      "jobTitle": "Example Workflow Job 1",
      "metadata": {
        "ScheduledJobId": "5fa2a3d17afcb9000179ad01",
        "ScheduledTaskId": "c95a93e0-b24f-44c3-a62f-7304c7ea7f2a"
      }
    },
    "created": "2020-11-04T12:53:42.266Z",
    "updated": "2020-11-04T12:53:46.676Z",
    "modified": "2020-11-04T12:53:47Z",
    "status": "Paused",
    "userId": "5d8cac54860b8100016b1987",
    "currentStep": {
      "singleStep": {
        "values": [
        ],
        "valueResourceIds": [
        ],
        "uniqueStepId": "0ffad266-89e0-4659-936b-c0f92d043c81",
        "previousUniqueStepId": "",
        "userId": "5d8cac54860b8103316b1987",
        "userName": "api.user1@example.com",
        "deviceId": null,
        "stepId": "41443154-f511-478b-bcca-849440421061",
        "stepNumber": 1,
        "stepTitle": "Enter Text",
        "stepDescription": "",
        "stepType": "Text",
        "connectionType": null,
        "started": "2020-11-04T12:53:42.687Z",
        "completed": null,
        "updated": "2020-11-04T12:53:46.664Z",
        "cancelled": null,
        "timeEvents": {
          "2020-11-04T12:53:46.664Z": 0
        },
        "note": null,
        "noteResourceFilenames": null,
        "coordinates": null,
        "parentStepId": null,
        "parentUniqueStepId": null,
        "parentStepTitle": null,
        "parentStepDescription": null,
        "metadata": {
          "tags": ""
        }
      },
      "reportStep": null
    },
    "abandonReason": null,
    "username": "api.user1@example.com",
```

```

    "workflowName": "Example Workflow"
  },
  "completedSteps": null,
  "cancelledSteps": null,
  "team": "apiteam",
  "hierarchyStack": null,
  "packetIds": null,
  "jobExclusion": [
  ],
  "excluded": false
},
{
  "jobId": "5f046ecf7df8a5000110cc54",
  "metadata": {
    "clientJobId": "f6ec68ae-a0b9-4725-aef4-9fb224be8e9d",
    "workflowId": "a454a699-173f-429e-b6cd-fe9d92d0a1er",
    "workflowVersionId": "c683d13a-4cda-4683-1276-e7fcfb9c1c52",
    "jobTitle": "Example Workflow Job 2",
    "metadata": {
    },
    "created": "2020-07-07T12:47:09.281Z",
    "updated": "2020-07-07T12:47:17.619Z",
    "modified": "2020-07-07T12:47:17Z",
    "status": "Completed",
    "userId": "5d8cac54860b8103416b1987",
    "currentStep": null,
    "abandonReason": null,
    "username": "api.user2@example.com",
    "workflowName": "Example Workflow"
  },
  "completedSteps": null,
  "cancelledSteps": null,
  "team": "testteam",
  "hierarchyStack": null,
  "packetIds": null,
  "jobExclusion": [
  ],
  "excluded": false
}
]

```

6.3. Get Snapshots of a Job

6.3.1. HTTP Request

```
GET https://gateway.flow.librestream.com/jobs/v1/snapshots/{client_job_id}
```

6.3.2. Authorization

Bearer Token

Policy [Job_ReadAll](#) (on page 26)

6.3.3. Headers

Key	Value
Authorization	Bearer {token}

6.3.4. Path Parameters

Parameter	Type
client_job_id	string

6.3.5. Response Body

6.3.5.1. JSON Representation

--

6.4. Get Job by Client Job Id

6.4.1. HTTP Request

```
GET https://gateway.flow.librestream.com/jobs/v1/{client_job_id}
```

6.4.2. Authorization

Bearer Token

Policy [Job_Read](#) (on page 26)

6.4.3. Headers

Key	Value
Authorization	Bearer {token}

6.4.4. Path Parameters

Parameter	Type
client_job_id	string

6.4.5. Response Body

6.4.5.1. JSON Representation

```
[
  {
    "jobId": "5fa2a458980724000149rq40",
    "metadata": {
      "clientJobId": "8a07b1c0-56ba-433d-8a24-adddgb3e150c",
      "workflowId": "a454a699-173f-429e-b6cd-fe9d92d0a1er",
      "workflowVersionId": "c683d13a-4cda-4683-1276-e7fcfb9c1c52",
      "jobTitle": "Example Workflow Job 1",
      "metadata": {
        "ScheduledJobId": "5fa2a3d17afcb9000179ad01",
        "ScheduledTaskId": "c95a93e0-b24f-44c3-a62f-7304c7ea7f2a"
      }
    },
    "created": "2020-11-04T12:53:42.266Z",
    "updated": "2020-11-04T12:53:46.676Z",
    "modified": "2020-11-04T12:53:47Z",
    "status": "Paused",
    "userId": "5d8cac54860b8100016b1987",
    "currentStep": {
      "singleStep": {
        "values": [
        ],
        "valueResourceIds": [
        ],
        "uniqueStepId": "0ffad266-89e0-4659-936b-c0f92d043c81",
        "previousUniqueStepId": "",
        "userId": "5d8cac54860b8103316b1987",
        "userName": "api.user1@example.com",
        "deviceId": null,
        "stepId": "41443154-f511-478b-bcca-849440421061",
        "stepNumber": 1,
        "stepTitle": "Enter Text",
        "stepDescription": "",
        "stepType": "Text",
        "connectionType": null,
        "started": "2020-11-04T12:53:42.687Z",
        "completed": null,
        "updated": "2020-11-04T12:53:46.664Z",
        "cancelled": null,
        "timeEvents": {
          "2020-11-04T12:53:46.664Z": 0
        },
        "note": null,
        "noteResourceFilenames": null,
        "coordinates": null,
        "parentStepId": null,
        "parentUniqueStepId": null,
        "parentStepTitle": null,
        "parentStepDescription": null,
        "metadata": {
          "tags": ""
        }
      },
      "reportStep": null
    },
    "abandonReason": null,
    "username": "api.user1@example.com",
```

```

    "workflowName": "Example Workflow"
  },
  "completedSteps": null,
  "cancelledSteps": null,
  "team": "apiteam",
  "hierarchyStack": null,
  "packetIds": null,
  "jobExclusion": [
  ],
  "excluded": false
}
]

```

6.5. Get Job by Job Id

6.5.1. HTTP Request

```
GET https://gateway.flow.librestream.com/jobs/v1/bson/{job_id}
```

6.5.2. Authorization

Bearer Token

Policy [Job_Read](#) (on page 26)

6.5.3. Headers

Key	Value
Authorization	Bearer {token}

6.5.4. Path Parameters

Parameter	Type
job_id	string

6.5.5. Response Body

6.5.5.1. JSON Representation

```

[
  {
    "jobId": "5fa2a458980724000149rq40",
    "metadata": {
      "clientJobId": "8a07b1c0-56ba-433d-8a24-adddgb3e150c",
      "workflowId": "a454a699-173f-429e-b6cd-fefd92d0a1er",
      "workflowVersionId": "c683d13a-4cda-4683-1276-e7fcfb9c1c52",

```

```

"jobTitle":"Example Workflow Job 1",
"metadata":{
  "ScheduledJobId":"5fa2a3d17afcb9000179ad01",
  "ScheduledTaskId":"c95a93e0-b24f-44c3-a62f-7304c7ea7f2a"
},
"created":"2020-11-04T12:53:42.266Z",
"updated":"2020-11-04T12:53:46.676Z",
"modified":"2020-11-04T12:53:47Z",
"status":"Paused",
"userId":"5d8cac54860b8100016b1987",
"currentStep":{
  "singleStep":{
    "values":[
    ],
    "valueResourceIds":[
    ],
    "uniqueStepId":"0ffad266-89e0-4659-936b-c0f92d043c81",
    "previousUniqueStepId":"",
    "userId":"5d8cac54860b8103316b1987",
    "userName":"api.user1@example.com",
    "deviceId":null,
    "stepId":"41443154-f511-478b-bcca-849440421061",
    "stepNumber":1,
    "stepTitle":"Enter Text",
    "stepDescription":"",
    "stepType":"Text",
    "connectionType":null,
    "started":"2020-11-04T12:53:42.687Z",
    "completed":null,
    "updated":"2020-11-04T12:53:46.664Z",
    "cancelled":null,
    "timeEvents":{
      "2020-11-04T12:53:46.664Z":0
    },
    "note":null,
    "noteResourceFileNames":null,
    "coordinates":null,
    "parentStepId":null,
    "parentUniqueStepId":null,
    "parentStepTitle":null,
    "parentStepDescription":null,
    "metadata":{
      "tags":""
    }
  },
  "reportStep":null
},
"abandonReason":null,
"username":"api.user1@example.com",
"workflowName":"Example Workflow"
},
"completedSteps":null,
"cancelledSteps":null,
"team":"apiteam",
"hierarchyStack":null,
"packetIds":null,
"jobExclusion":[
],
"excluded":false

```

```

},
{
  "jobId": "5f046ecf7df8a5000110cc54",
  "metadata": {
    "clientJobId": "f6ec68ae-a0b9-4725-aef4-9fb224be8e9d",
    "workflowId": "a454a699-173f-429e-b6cd-fe9d92d0a1er",
    "workflowVersionId": "c683d13a-4cda-4683-1276-e7fcfb9c1c52",
    "jobTitle": "Example Workflow Job 2",
    "metadata": {
    },
    "created": "2020-07-07T12:47:09.281Z",
    "updated": "2020-07-07T12:47:17.619Z",
    "modified": "2020-07-07T12:47:17Z",
    "status": "Completed",
    "userId": "5d8cac54860b8103416b1987",
    "currentStep": null,
    "abandonReason": null,
    "username": "api.user2@example.com",
    "workflowName": "Example Workflow"
  },
  "completedSteps": null,
  "cancelledSteps": null,
  "team": "testteam",
  "hierarchyStack": null,
  "packetIds": null,
  "jobExclusion": [
  ],
  "excluded": false
}
]

```

6.6. Get My Jobs

6.6.1. HTTP Request

```
GET https://gateway.flow.librestream.com/jobs/v1/my/?{skip}&{limit}
```

6.6.2. Authorization

Bearer Token

Policy [Job_Read \(on page 26\)](#)

6.6.3. Headers

Key	Value
Authorization	Bearer {token}

6.6.4. Query Parameters

Parameter	Type
skip	integer
limit	integer

NB: Skip - To skip first {skip} number of jobs , Limit - To limit the results to {limit} number of jobs. Default limit is 20.

6.6.5. Response Body

6.6.5.1. JSON Representation

```
[
  {
    "jobId": "5fa2a458980724000149rq40",
    "metadata": {
      "clientJobId": "8a07b1c0-56ba-433d-8a24-adddgb3e150c",
      "workflowId": "a454a699-173f-429e-b6cd-fe9d92d0a1er",
      "workflowVersionId": "c683d13a-4cda-4683-1276-e7fcfb9c1c52",
      "jobTitle": "Example Workflow Job 1",
      "metadata": {
        "ScheduledJobId": "5fa2a3d17afcb9000179ad01",
        "ScheduledTaskId": "c95a93e0-b24f-44c3-a62f-7304c7ea7f2a"
      }
    },
    "created": "2020-11-04T12:53:42.266Z",
    "updated": "2020-11-04T12:53:46.676Z",
    "modified": "2020-11-04T12:53:47Z",
    "status": "Paused",
    "userId": "5d8cac54860b8100016b1987",
    "currentStep": {
      "singleStep": {
        "values": [
        ],
        "valueResourceIds": [
        ],
        "uniqueStepId": "0ffad266-89e0-4659-936b-c0f92d043c81",
        "previousUniqueStepId": "",
        "userId": "5d8cac54860b8103316b1987",
        "userName": "api.user1@example.com",
        "deviceId": null,
        "stepId": "41443154-f511-478b-bcca-849440421061",
        "stepNumber": 1,
        "stepTitle": "Enter Text",
        "stepDescription": "",
        "stepType": "Text",
        "connectionType": null,
        "started": "2020-11-04T12:53:42.687Z",
        "completed": null,
        "updated": "2020-11-04T12:53:46.664Z",
        "cancelled": null,
        "timeEvents": {
          "2020-11-04T12:53:46.664Z": 0
        }
      }
    }
  }
]
```

```

    },
    "note": null,
    "noteResourceFileNames": null,
    "coordinates": null,
    "parentStepId": null,
    "parentUniqueStepId": null,
    "parentStepTitle": null,
    "parentStepDescription": null,
    "metadata": {
      "tags": ""
    }
  },
  "reportStep": null
},
"abandonReason": null,
"username": "api.user1@example.com",
"workflowName": "Example Workflow"
},
"completedSteps": null,
"cancelledSteps": null,
"team": "apiteam",
"hierarchyStack": null,
"packetIds": null,
"jobExclusion": [
],
"excluded": false
}
]

```

6.7. Abort a Job

6.7.1. HTTP Request

POST https://gateway.flow.librestream.com/jobs/v1/{client_job_id}/abort

6.7.2. Authorization

Bearer Token

Policy [Job_Read \(on page 26\)](#)

6.7.3. Headers

	Key	Value
Authorization		Bearer {token}

6.7.4. Path Parameters

Parameter	Type
client_job_id	string

NB: Only incomplete jobs can be aborted.

6.8. Delete a Job

6.8.1. HTTP Request

```
DEL https://gateway.flow.librestream.com/jobs/v1/{client_job_id}
```

6.8.2. Authorization

Bearer Token

Policy [Job_Modify \(on page 25\)](#)

6.8.3. Headers

Key	Value
Authorization	Bearer {token}

6.8.4. Path Parameters

Parameter	Type
client_job_id	string

6.9. Assign User to a Job

6.9.1. HTTP Request

```
POST https://gateway.flow.librestream.com/jobs/v1/{client_job_id}/assign/{user_id}
```

6.9.2. Authorization

Bearer Token

Policy [Job_Read \(on page 26\)](#)

6.9.3. Headers

Key	Value
Authorization	Bearer {token}

6.9.4. Path Parameters

Parameter	Type
client_job_id	string
user_id	string

NB: Aborted and Completed jobs can't be assigned to another user.

6.10. Exclude a Job

6.10.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/jobs/v1/{client_job_id}
```

6.10.2. Authorization

Bearer Token

Policy [Job_ReadAll](#) (on page 26)

6.10.3. Headers

Key	Value
Authorization	Bearer {token}

6.10.4. Path Parameters

Parameter	Type
client_job_id	string

6.10.5. Request Body

6.10.5.1. JSON Representation

```
{
  "userId": "5d8cac54860b8103316b1987",
  "reason": "Api Exclude Job Test",
  "isExcluded": "true"
}
```

6.11. Add Metadata to a Job

6.11.1. HTTP Request

POST `https://gateway.flow.librestream.com/jobs/v1/{client_job_id}/metadata`

6.11.2. Authorization

Bearer Token

Policy [Job_Modify \(on page 25\)](#)

6.11.3. Headers

Key	Value
Authorization	Bearer {token}

6.11.4. Path Parameters

Parameter	Type
client_job_id	string

6.11.5. Request Body

6.11.5.1. JSON Representation

```
{
  "key": "value"
}
```

6.12. Create a Job

6.12.1. HTTP Request

```
POST https://gateway.flow.librestream.com/jobs/v1
```

6.12.2. Authorization

Bearer Token

Policy [Workflow_Execute](#) (on page 33)

6.12.3. Headers

	Key	Value
Authorization		Bearer {token}

6.12.4. Request Body

6.12.4.1. JSON Representation

```
{
  "metadata": {
    "clientJobId": "cb3dc8dc-4162-47c4-af05-3575034f99e5",
    "workflowId": "b62f2507-b4e7-4248-83d3-29c03a158fbe",
    "workflowVersionId": "2831f333-af02-4d89-875d-0bceadeb16ed",
    "jobTitle": "Api Test 2",
    "metadata": {
      "Key1": "V1",
    },
    "created": "2021-10-01T12:48:12.215Z",
    "updated": "2021-10-01T12:48:13.824Z",
    "modified": "2021-10-01T12:48:14Z",
    "status": "Completed",
    "userId": "5d8cac54860b8103316b1987",
    "currentStep": null,
    "abandonReason": null,
    "username": "api.user1@example.com",
    "workflowName": "Example Workflow"
  },
  "completedSteps": [
    {
      "singleStep": {
        "values": [
        ],
        "valueResourceIds": [
        ],
        "uniqueStepId": "c025f569-8148-4a75-a2e8-a9bc19da911d",
        "previousUniqueStepId": "",
        "userId": "5d8cac54860b8103316b1987",
        "userName": "api.user1@example.com",
        "deviceId": null,
        "stepId": "ec7eee0d-5115-4141-847a-64c28694dde2",
      }
    }
  ]
}
```

```

        "stepNumber":1,
        "stepTitle":"Instruction",
        "stepDescription":"",
        "stepType":"ConfirmStep",
        "connectionType":null,
        "started":"2021-10-01T12:48:12.265Z",
        "completed":"2021-10-01T12:48:13.8Z",
        "updated":"2021-10-01T12:48:13.8Z",
        "cancelled":null,
        "timeEvents":null,
        "note":null,
        "noteResourceFileNames":null,
        "coordinates":null,
        "parentStepId":null,
        "parentUniqueStepId":null,
        "parentStepTitle":null,
        "parentStepDescription":null,
        "metadata":{
            "tags":""
        }
    },
    "reportStep":null
}],
"cancelledSteps":null,
"team":"apiteam",
"hierarchyStack":null,
"packetIds":null,
"jobExclusion":[
],
"excluded":false
}

```

6.12.5. Response Body

6.12.5.1. JSON Representation

```

{
  "jobId":"615704cbb249fb12315fcfbc",
  "metadata":{
    "clientJobId":"cb3dc8dc-4162-47c4-af05-3575034f99e5",
    "workflowId":"b62f2507-b4e7-4248-83d3-29c03a158fbe",
    "workflowVersionId":"2831f333-af02-4d89-875d-0bceadeb16ed",
    "jobTitle":"Api Test 2",
    "metadata":{
      "Key1":"V1",
    },
  },
  "created":"2021-10-01T12:48:12.215Z",
  "updated":"2021-10-01T12:48:13.824Z",
  "modified":"2021-10-01T12:48:14Z",
  "status":"Completed",
  "userId":"5d8cac54860b8103316b1987",
  "currentStep":null,
  "abandonReason":null,
  "username":"api.user1@example.com",
  "workflowName":"Example Workflow"
},

```

```

"completedSteps":[
  {
    "singleStep":{
      "values":[
      ],
      "valueResourceIds":[
      ],
      "uniqueStepId":"c025f569-8148-4a75-a2e8-a9bc19da911d",
      "previousUniqueStepId":"",
      "userId":"5d8cac54860b8103316b1987",
      "userName":"api.user1@example.com",
      "deviceId":null,
      "stepId":"ec7eee0d-5115-4141-847a-64c28694dde2",
      "stepNumber":1,
      "stepTitle":"Instruction",
      "stepDescription":"",
      "stepType":"ConfirmStep",
      "connectionType":null,
      "started":"2021-10-01T12:48:12.265Z",
      "completed":"2021-10-01T12:48:13.8Z",
      "updated":"2021-10-01T12:48:13.8Z",
      "cancelled":null,
      "timeEvents":null,
      "note":null,
      "noteResourceFileNames":null,
      "coordinates":null,
      "parentStepId":null,
      "parentUniqueStepId":null,
      "parentStepTitle":null,
      "parentStepDescription":null,
      "metadata":{
        "tags":""
      }
    },
    "reportStep":null
  }],
"cancelledSteps":null,
"team":"apiteam",
"hierarchyStack":null,
"packetIds":null,
"jobExclusion":[
],
"excluded":false
}

```

6.13. Abort Jobs for an User

6.13.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/jobs/v1/abort/?{userId}
```

6.13.2. Authorization

Bearer Token

6.13.3. Headers

Key	Value
Authorization	Bearer {token}

6.13.4. Query Parameters

Parameter	Type
userId	string

NB: Only incomplete jobs can be aborted. When an User Id is not specified it would delete all users/team's incomplete jobs.

6.14. Basic Sync

6.14.1. Overview

BasicSync endpoint can be used in 3 ways.

1. Creates a job if one with the current client job id doesn't exist.
2. Updates a job if the job exists already.
3. Aborts the job if the job is being updated and the status is 'Aborted'.

6.14.2. HTTP Request

```
POST https://gateway.flow.librestream.com/jobs/v1/basicSync
```

6.14.3. Authorization

Bearer Token

Policy [Workflow_Execute \(on page 33\)](#)

6.14.4. Headers

Key	Value
Authorization	Bearer {token}

6.14.5. Request Body

6.14.5.1. JSON Representation

```
{
  "metadata":{
    "clientJobId":"cb3dc8dc-4162-47c4-af05-3575034f99e5",
    "workflowId":"b62f2507-b4e7-4248-83d3-29c03a158fbe",
    "workflowVersionId":"2831f333-af02-4d89-875d-0bceadeb16ed",
    "jobTitle":"Api Test 2",
    "metadata":{
      "Key1":"V1",
    },
    "created":"2021-10-01T12:48:12.215Z",
    "updated":"2021-10-01T12:48:13.824Z",
    "modified":"2021-10-01T12:48:14Z",
    "status":"Completed",
    "userId":"5d8cac54860b8103316b1987",
    "currentStep":null,
    "abandonReason":null,
    "username":"api.user1@example.com",
    "workflowName":"Example Workflow"
  },
  "completedSteps":[
    {
      "singleStep":{
        "values":[
        ],
        "valueResourceIds":[
        ],
        "uniqueStepId":"c025f569-8148-4a75-a2e8-a9bc19da911d",
        "previousUniqueStepId":"",
        "userId":"5d8cac54860b8103316b1987",
        "userName":"api.user1@example.com",
        "deviceId":null,
        "stepId":"ec7eee0d-5115-4141-847a-64c28694dde2",
        "stepNumber":1,
        "stepTitle":"Instruction",
        "stepDescription":"",
        "stepType":"ConfirmStep",
        "connectionType":null,
        "started":"2021-10-01T12:48:12.265Z",
        "completed":"2021-10-01T12:48:13.8Z",
        "updated":"2021-10-01T12:48:13.8Z",
        "cancelled":null,
        "timeEvents":null,
        "note":null,
        "noteResourceFilenames":null,
        "coordinates":null,
        "parentStepId":null,
        "parentUniqueStepId":null,
        "parentStepTitle":null,
        "parentStepDescription":null,
        "metadata":{
          "tags":""
        }
      },
      "reportStep":null
    },
  ],
  "cancelledSteps":null,
```

```
"team":"apiteam",
"hierarchyStack":null,
"packetIds":null,
"jobExclusion":[
],
"excluded":false
}
```

6.15. Update a Job

6.15.1. HTTP Request

POST https://gateway.flow.librestream.com/jobs/v1/{client_job_id}

6.15.2. Authorization

Bearer Token

Policy [Job_Modify \(on page 25\)](#)

6.15.3. Headers

Key	Value
Authorization	Bearer {token}

6.15.4. Path Parameters

Parameter	Type
client_job_id	string

6.15.5. Request Body

6.15.5.1. JSON Representation

```
{
  "metadata":{
    "clientJobId":"cb3dc8dc-4162-47c4-af05-3575034f99e5",
    "workflowId":"b62f2507-b4e7-4248-83d3-29c03a158fbe",
    "workflowVersionId":"2831f333-af02-4d89-875d-0bceadeb16ed",
    "jobTitle":"Api Test 2 - Updated name",
    "metadata":{
      "Key1":"V1",
    },
    "created":"2021-10-01T12:48:12.215Z",
    "updated":"2021-10-01T12:48:13.824Z",
    "modified":"2021-10-01T12:48:14Z",
    "status":"Paused",
    "userId":"5d8cac54860b8103316b1987",
    "currentStep":null,
    "abandonReason":null,
    "username":"api.user1@example.com",
    "workflowName":"Example Workflow"
  },
}
```

7. Report Generator

Report Generator can be managed via the APIs

Related information

[Generate a Report \(on page 107\)](#)

[Generate Report by Report Template Id \(on page 108\)](#)

7.1. Generate a Report

7.1.1. HTTP Request

```
POST https://gateway.flow.librestream.com/reportgenerator/v1/generate?{jobId}
```

7.1.2. Authorization

Bearer Token

Policy [Reports_Read \(on page 28\)](#)

7.1.3. Headers

Key	Value
Authorization	Bearer {token}

7.1.4. Query Parameters

Parameter	Type
jobId	string

7.1.5. Response Body

7.1.5.1. JSON Representation

```
{
  "jobId": "615af46c3396ba3001e40675",
  "status": "Completed",

  "reportUrl": "https://
wfp27dev.blob.core.windows.net/reportpdfs/apiteam/615af46c3496ba0301c40675_5eb113b42faf0
f254b192de7?sv=2019-02-02&sr=b&sig=y068gzJ0upmCwSwwXD7EMh0Qggz9IRpjTTCsS0tq5nE%3D&st=202
1-12-09T13%3A41%3A42Z&se=2021-12-09T14%3A46%3A42Z&sp=r"
}
```

7.2. Generate Report by Report Template Id

7.2.1. HTTP Request

```
POST
https://gateway.flow.librestream.com/reportgenerator/v1/generate/{reportTemplateId}?{jobId}
```

7.2.2. Authorization

Bearer Token

Policy [Reports_Read](#) (*on page 28*)

7.2.3. Headers

Key	Value
Authorization	Bearer {token}

7.2.4. Query Parameters

Parameter	Type
jobId	string

7.2.5. Path Parameters

Parameter	Type
reportTemplateId	string

7.2.6. Response Body

7.2.6.1. JSON Representation

```
{
  "jobId": "615af46c3396ba2001e40675",
  "status": "Completed",

  "reportUrl": "https://
wfp27dev.blob.core.windows.net/reportpdfs/apiteam/615af46c3396ba0301c40675_5eb113b42faf0
f254b192de7?sv=2019-02-02&sr=b&sig=y068gzJ0upmCwSwwXD7EMh0Qggz9IRpjTTCsS0tq5nE%3D&st=202
1-12-09T13%3A41%3A42Z&se=2021-12-09T14%3A46%3A42Z&sp=r"
}
```

8. Reports

Report Generator can be managed via the APIs

Related information

[Get PDF by Client Job Id](#) (on page 109)

8.1. Get PDF by Client Job Id

8.1.1. HTTP Request

```
GET https://gateway.flow.librestream.com/reports/v1/generate/{clientId}
```

8.1.2. Authorization

Bearer Token

Policy [Reports_Read](#) (on page 28)

8.1.3. Headers

Key	Value
Authorization	Bearer {token}

8.1.4. Query Parameters-1234

Parameter	Type
getUrlOnly	boolean

8.1.5. Path Parameters

Parameter	Type
clientId	string

8.1.6. Response Body

8.1.6.1. JSON Representation

```
https://wfp27dev.blob.core.windows.net/reportpdfs/testteam/615af46c3e96ba0001c4067_5eb113b42faf0f254b092de7?sv=2019-02-02&sr=b&sig=%2FYxBFVIs62iHEv73Aw9kzLZ9pJmirUPWHLsngQQc7Po%3D&st=2021-12-10T12%3Aw8%3A23Z&se=2021-12-10T13%3A13%3A23Z&sp=r
```


9. Report Templates

Report Templates can be managed via the APIs

Related information

[Get All Report Templates \(on page 111\)](#)

[Get Report Template \(on page 113\)](#)

[Get Report Template Names \(on page 114\)](#)

[Get Default Template \(on page 115\)](#)

[Delete Report Template \(on page 115\)](#)

[Set Default Report Template \(on page 116\)](#)

[Unset Default Report Template \(on page 117\)](#)

[Create Report Template \(on page 117\)](#)

[Update Template Metadata \(on page 118\)](#)

[Update Report Template \(on page 119\)](#)

9.1. Get All Report Templates

9.1.1. HTTP Request

```
GET https://gateway.flow.librestream.com/reporttemplates/v1
```

9.1.2. Authorization

Bearer Token

Policy [Reports_Create \(on page 27\)](#)

9.1.3. Headers

Key	Value
Authorization	Bearer {token}

9.1.4. Response Body

9.1.4.1. JSON Representation

```
[
  {
    "reportTemplateId": "5ec4f0fde9efaa03011a07bc",
    "name": "Example Report 1",
    "description": "To demo report templates apis",
    "template": {
      "templateId": "601291dd90a363400112c636",
      "templateBodyFile": {
```

```

        "name": "Body template",
        "uri": "1e74cc05-3304-497b-9690-35481cd61149",
        "contentType": "text/html",

        "downloadUrl": "https://
wfp27dev.blob.core.windows.net/reporttemplates/testteam/1e74cc05-4304-497b-9690-35481cd6
1349?sv=2019-02-02&sr=b&sig=AKhrBbbunmVDpI%2BEtULITA39HZ5G5j0ycMP3%2FGaHfek%3D&st=2021-1
0-08T07%3A10%3A11Z&se=2021-10-08T08%3A15%3A11Z&sp=r"}},
        "templateHeaderFile": {
            "name": "Header template",
            "uri": "f42d00af-b807-4d57-8aba-124e2fad80cf",
            "contentType": "text/html",

            "downloadUrl": "https://
wfp27dev.blob.core.windows.net/reporttemplates/testteam/f42e40af-b217-4d57-8aba-124e2fad
80cf?sv=2019-02-02&sr=b&sig=Z7%2B3hKXQFIUioVZKN4Gz5bQa5ZfVsJlKbEdxT4t7VzI%3D&st=2021-10-
08T07%3A10%3A11Z&se=2021-10-08T08%3A15%3A11Z&sp=r"}},
            "uploadedDate": "2021-01-28T10:28:45Z",
            "userId": "5d8cacd378411f0321e71db0"},
        "lastUpdated": "2021-01-28T10:28:45Z",
        "options": {
            "isLandscape": false,
            "title": "Example Report 1",
            "headerHeight": 48,
            "hideFooter": false},
        "isDefault": false,
        "isArchived": false,
        "metadata": {}
    },
    {
        "reportTemplateId": "5f036028e031440091c9a073",
        "name": "Example Report 2",
        "description": null,
        "template": {
            "templateId": "5f0432c4e031440001c9c8e6",
            "templateBodyFile": {
                "name": "Body template",
                "uri": "1c0d47f0-1e6f-41f4-893d-0233df06a332",
                "contentType": "text/html",

                "downloadUrl": "https://
wfp27dev.blob.core.windows.net/reporttemplates/testteam/1c0d47f0-1e6f-41f4-893d-0233df06
a332?sv=2019-02-02&sr=b&sig=S%2FbmcGnJm9gUJ7%2F4ZzLX1J7sCwzh4ca%2BxhmN5EEaV5c%3D&st=2021
-10-08T07%3A10%3A11Z&se=2021-10-08T08%3A15%3A11Z&sp=r"}},
                "templateHeaderFile": {
                    "name": "Header template",
                    "uri": "46e29592-66f1-468b-9cad-cac35be1a0e0",
                    "contentType": "text/html",

                    "downloadUrl": "https://
wfp27dev.blob.core.windows.net/reporttemplates/testteam/46e22592-66f1-468b-9cad-cace5be1
a0e0?sv=2019-02-02&sr=b&sig=RhrtxbjJzi8klOHjkyTLB7ug%2BQnfz%2BsD28l3p%2FzFhpY%3D&st=2021
-10-08T07%3A10%3A11Z&se=2021-10-08T08%3A15%3A11Z&sp=r"}},
                    "uploadedDate": "2020-07-07T08:31:00Z",
                    "userId": "5e21df72e7fb2433011f12fd"},
                "lastUpdated": "2020-07-14T07:27:24Z",
                "options": {
                    "isLandscape": false,
                    "title": "Example Report 2",

```

```
    "headerHeight":48,  
    "hideFooter":false},  
    "isDefault":false,  
    "isArchived":false,  
    "metadata":{  
      "jobId":"5f0d5ac0bafw9b00013c2495",  
      "workflowId":"a1d4234f-f67d-49f6-92e5-2fc0c4d08643"}  
  }  
]
```

9.2. Get Report Template

9.2.1. HTTP Request

GET https://gateway.flow.librestream.com/reporttemplates/v1/content/{reportTemplate_id}

9.2.2. Authorization

Bearer Token

Policy [Reports_Create](#) (on page 117)

9.2.3. Headers

Key	Value
Authorization	Bearer {token}

9.2.4. Path Parameters

Parameter	Type
reportTemplate_id	string

9.2.5. Response Body

9.2.5.1. JSON Representation

```
{
  "reportTemplateId": "615fff0b1b4ee200018fe357",
  "name": "Report Template Example 1",
  "description": null,
  "headerContent": "<header>\n</header>",
  "bodyContent": "<!doctype html>\n<html lang=\"en\">\n  <head>\n<meta
charset=\"utf-8\">\n
                <style type=\"text/css\"></style>\n  </head>\n  <body>
                <h1> Api Testing Report Template </h1></body>\n</html>",
  "lastUpdated": "2021-10-08T13:19:14Z",
  "userId": "5d8cac54860b8100016b1987",
  "options": {
    "isLandscape": false,
    "title": "Report Template Example 1",
    "headerHeight": 48,
    "hideFooter": false},
  "isDefault": false,
  "isArchived": false,
  "metadata": {}
}
```

9.3. Get Report Template Names

9.3.1. HTTP Request

```
GET https://gateway.flow.librestream.com/reporttemplates/v1/names
```

9.3.2. Authorization

Bearer Token

Policy [Reports_Create](#) (on page 27)

9.3.3. Headers

Key	Value
Authorization	Bearer {token}

9.3.4. Response Body

9.3.4.1. JSON Representation

```
[
  {
    "reportTemplateId": "5ec4f0fde9efaa00011a07bc",
    "name": "Report Template Example 1",
    "description": null,
    "templateVersions": null,
    "teamName": null,
    "lastUpdated": "0001-01-01T00:00:00",
    "template": null,
    "options": null,
    "isArchived": false,
    "metadata": {}
  },
  {
    "reportTemplateId": "5f036028e031440001c9a073",
    "name": "Report Template Example 2",
    "description": null,
    "templateVersions": null,
    "teamName": null,
    "lastUpdated": "0001-01-01T00:00:00",
    "template": null,
    "options": null,
    "isArchived": false,
    "metadata": {}
  }
]
```

9.4. Get Default Template

9.4.1. HTTP Request

```
GET https://gateway.flow.librestream.com/reporttemplates/v1/default
```

9.4.2. Authorization

Bearer Token

Policy [Reports_Create](#) (on page 27)

9.4.3. Headers

Key	Value
Authorization	Bearer {token}

9.5. Delete Report Template

9.5.1. HTTP Request

```
DEL https://gateway.flow.librestream.com/reporttemplates/v1/{reportTemplate_id}
```

9.5.2. Authorization

Bearer Token

Policy [Reports_Create](#) (on page 27)

9.5.3. Headers

Key	Value
Authorization	Bearer {token}

9.5.4. Path Parameters

Parameter	Type
reportTemplate_id	string

9.6. Set Default Report Template

9.6.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/reporttemplates/v1/default/{reportTemplate_id}
```

9.6.2. Authorization

Bearer Token

Policy [Reports_Create](#) (on page 27)

9.6.3. Headers

Key	Value
Authorization	Bearer {token}

9.6.4. Path Parameters

Parameter	Type
reportTemplate_id	string

9.7. Unset Default Report Template

9.7.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/reporttemplates/v1/default/unset
```

9.7.2. Authorization

Bearer Token

Policy [Reports_Create \(on page 27\)](#)

9.7.3. Headers

Key	Value
Authorization	Bearer {token}

9.8. Create Report Template

9.8.1. HTTP Request

```
POST https://gateway.flow.librestream.com/reporttemplates/v1/content
```

9.8.2. Authorization

Bearer Token

Policy [Reports_Create \(on page 27\)](#)

9.8.3. Headers

Key	Value
Authorization	Bearer {token}

9.8.4. Request Body

9.8.4.1. JSON Representation

```
{
  "name": "APi Template Example 1",
  "headerContent": "<header>\n</header>",
  "bodyContent": "<!doctype html>\n<html lang=\"en\">\n\n  <head>\n    <meta
charset=\"utf-8\">\n
      <style type=\"text/css\">\n    </style>\n  </head>\n\n  <body>\n\n
</body>\n\n</html>",
  "options": {"headerHeight": 48, "title": "APi Template Example 1"}
}
```

9.8.5. Response Body

9.8.5.1. JSON Representation

```
{
  "reportTemplateId": "61652831013a4200010e5405",
  "name": "APi Template Example 1",
  "description": null,
  "headerContent": "<header>\n</header>",
  "bodyContent": "<!doctype html>\n<html lang=\"en\">\n\n  <head>\n    <meta
charset=\"utf-8\">\n
      <style type=\"text/css\">\n    </style>\n  </head>\n\n  <body>\n\n
</body>\n\n</html>",
  "lastUpdated": "0001-01-01T00:00:00",
  "userId": null,
  "options": {
    "isLandscape": false,
    "title": "APi Template Example 1",
    "headerHeight": 48,
    "hideFooter": false
  },
  "isDefault": false,
  "isArchived": false,
  "metadata": {
  }
}
```

9.9. Update Template Metadata

9.9.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/reporttemplates/v1/metadata/{reportTemplate_id}
```

9.9.2. Authorization

Bearer Token

9.9.3. Headers

Key	Value
Authorization	Bearer {token}

9.9.4. Path Parameters

Parameter	Type
reportTemplate_id	string

9.9.5. Request Body

9.9.5.1. JSON Representation

```
{
  "key": "value"
}
```

9.10. Update Report Template

9.10.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/reporttemplates/v1/content/{reportTemplate_id}
```

9.10.2. Authorization

Bearer Token

Policy [Reports_Create](#) (on page 27)

9.10.3. Headers

Key	Value
Authorization	Bearer {token}

9.10.4. Path Parameters

Parameter	Type
reportTemplate_id	string

9.10.5. Request Body

9.10.5.1. JSON Representation

```
{
  "reportTemplateId": "61652831017a4200010e5405",
  "name": "APi Template Example 1",
  "description": null,
  "headerContent": "<header>\n</header>",
  "bodyContent": "<!doctype html>\n<html lang=\"en\">\n  <head>\n    <meta
charset=\"utf-8\">\n
    <style type=\"text/css\"></style>\n  </head>\n  <body>
    <h1> Updated Via Api Call </h1>  </body>\n</html>",
  "lastUpdated": "2021-10-12T06:29:23Z",
  "userId": "5d8cac54860b8100016b1987",
  "options": {"isLandscape": false,
    "title": "APi Template Example 1",
    "headerHeight": 48,
    "hideFooter": false},
  "isDefault": false,
  "isArchived": false,
  "metadata": {}
}
```

10. Scheduler

Scheduler can be managed via the APIs

Related information

[Get All Scheduled Jobs \(on page 121\)](#)

[Get Scheduled Job \(on page 123\)](#)

[Get My Scheduled Jobs \(on page 124\)](#)

[Update the status of a Task in a Scheduled Job \(on page 125\)](#)

[Update the Status of a Scheduled Job \(on page 126\)](#)

[Update Assigned User for a Job \(on page 127\)](#)

[Delete a Scheduled Job \(on page 128\)](#)

[Create a Scheduled Job \(on page 128\)](#)

10.1. Get All Scheduled Jobs

10.1.1. HTTP Request

```
GET https://gateway.flow.librestream.com/scheduledjobs/v1
```

10.1.2. Authorization

Bearer Token

Policy [Schedule_ReadAll \(on page 29\)](#)

10.1.3. Headers

Key		Value
Authorization		Bearer {token}

10.1.4. Response Body

10.1.4.1. JSON Representation

```
[
  {
    "jobId": "5f365519a2ed5a2301f2d303",
    "created": "2021-08-14T09:10:49.636Z",
    "updated": "2021-08-14T09:10:49.636Z",
    "state": "Scheduled",
    "name": "Scheduled Job1",
    "description": "Scheduled Job1 - WF1",
    "priority": false,
    "dueByDate": null,
    "assignedUserIds": [
    ],
    "tasks":
    [
      {
        "taskType": "Workflow",
        "taskId": "707c5cb6-8171-4232-baca-f7c94dc42ea9",
        "artifactId": null,
        "jobId": null,
        "clientJobId": null,
        "updated": null,
        "modified": null,
        "state": 0
      }
    ]
  },
  {
    "jobId": "601170d37e97f340011ca1e3",
    "created": "2021-01-27T13:55:31.045Z",
    "updated": "2021-01-27T13:58:13.452Z",
    "state": "Completed",
    "name": "Scheduled Job2",
    "description": "Scheduled Job1 - WF2",
    "priority": true,
    "dueByDate": "2021-01-27T15:30:00Z",
    "assignedUserIds": [
      "5d8cacd378111f0001e71db0"
    ],
    "tasks":
    [
      {
        "taskType": "Workflow",
        "taskId": "610d8443-fdac-443c-bca5-52c761a56561",
        "artifactId": "012ee471-9954-4b53-93ac-1062e43bffa3",
        "jobId": "6011713d19863200010ed4b5",
        "clientJobId": "7aa2546f-be00-45fd-94fa-d3ccfbe22fdb",
        "updated": "2021-01-27T13:58:13.363Z",
        "modified": "2021-01-27T13:58:13.366Z",
        "state": 3
      }
    ]
  }
]
```

10.2. Get Scheduled Job

10.2.1. HTTP Request

```
GET https://gateway.flow.librestream.com/scheduledjobs/v1/{scheduled_job_id}
```

10.2.2. Authorization

Bearer Token

Policy [Schedule_Read](#) (on page 29)

10.2.3. Headers

Key	Value
Authorization	Bearer {token}

10.2.4. Path Parameters

Parameter	Type
scheduled_job_id	string

10.2.5. Response Body

10.2.5.1. JSON Representation

```
{
  "jobId": "615c2057c5722f000168e37f",
  "created": "2021-10-05T09:52:23.341Z",
  "updated": "2021-10-05T09:52:23.341Z",
  "state": "Scheduled",
  "name": "API Scheduler Example Job 1",
  "description": "To Demo Scheduler Apis",
  "priority": true,
  "dueByDate": "2021-10-05T22:59:00Z",
  "assignedUserIds": [
    "5d8cac54860b81ee016b1987"],
  "tasks": [
    {
      "taskType": "Workflow",
      "taskId": "d7020dd1-8ce9-42ce-90f2-8d2f7f734c20",
      "artifactId": "b62f2507-b4e7-4248-83d3-29c03b158fbe",
      "jobId": null,
      "clientJobId": null,
      "updated": null,
      "modified": null,
      "state": 0
    }
  ]
}
```

NB: Task State

1. 1 - Active
2. 2 - Paused
3. 3 - Completed
4. 4 - Abandoned
5. 0 - Not started

10.3. Get My Scheduled Jobs

10.3.1. HTTP Request

```
GET https://gateway.flow.librestream.com/scheduledjobs/v1/my
```

10.3.2. Authorization

Bearer Token

Policy [Schedule_Read \(on page 29\)](#)

10.3.3. Headers

Key	Value
Authorization	Bearer {token}

10.3.4. Response Body

10.3.4.1. JSON Representation

```
{
  "jobId": "615c2057c5722f000168e37f",
  "created": "2021-10-05T09:52:23.341Z",
  "updated": "2021-10-05T09:52:23.341Z",
  "state": "Scheduled",
  "name": "API Scheduler Example Job 1",
  "description": "To Demo Scheduler Apis",
  "priority": true,
  "dueByDate": "2021-10-05T22:59:00Z",
  "assignedUserIds": [
    "5d8cac54860b81ee016b1987"],
  "tasks": [
    {
      "taskType": "Workflow",
      "taskId": "d7020dd1-8ce9-42ce-90f2-8d2f7f734c20",
      "artifactId": "b62f2507-b4e7-4248-83d3-29c03b158fbe",
      "jobId": null,
      "clientJobId": null,
      "updated": null,
      "modified": null,
      "state": 0
    }
  ]
}
```

10.4. Update the status of a Task in a Scheduled Job

10.4.1. HTTP Request

```
PUT
https://gateway.flow.librestream.com/scheduledjobs/v1/{scheduled_job_id}/tasks/{task_id}/status
```

10.4.2. Authorization

Bearer Token

Policy [Schedule_Read](#) (on page 29)

10.4.3. Headers

Key	Value
Authorization	Bearer {token}

10.4.4. Path Parameters

Parameter	Type
scheduled_job_id	string
task_id	string

10.4.5. Request Body

10.4.5.1. JSON Representation

```
{
  "clientId": "c0bda7f7-0a1f-400c-8276-cef1c4ae75ea",
  "state": "Paused",
  "updated": "2021-10-06T07:24:52.44"
}
```

10.5. Update the Status of a Scheduled Job

10.5.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/scheduledjobs/v1/{scheduled_job_id}/status
```

10.5.2. Authorization

Bearer Token

Policy [Schedule_Read \(on page 29\)](#)

10.5.3. Headers

Key	Value
Authorization	Bearer {token}

10.5.4. Path Parameters

Parameter	Type
scheduled_job_id	string

10.5.5. Request Body

10.5.5.1. JSON Representation

```
{
  "state": "Completed",
  "updated": "2021-10-06T07:24:52.44"
}
```

NB: A job can't be completed until all tasks in a job are completed.

10.6. Update Assigned User for a Job

10.6.1. HTTP Request

```
PUT
https://gateway.flow.librestream.com/scheduledjobs/v1/{scheduled_job_id}/assignedUsers
```

10.6.2. Authorization

Bearer Token

Policy [Schedule_Admin](#) (on page 28)

10.6.3. Headers

Key	Value
Authorization	Bearer {token}

10.6.4. Path Parameters

Parameter	Type
scheduled_job_id	string

10.6.5. Request Body

10.6.5.1. JSON Representation

```
{
  "assignedUserIds": ["5d91f997ccfe6a0001dc9dc1"]
}
```

10.7. Delete a Scheduled Job

10.7.1. HTTP Request

```
DEL https://gateway.flow.librestream.com/scheduledjobs/v1/{scheduled_job_id}
```

10.7.2. Authorization

Bearer Token

Policy [Schedule_Admin](#) (on page 28)

10.7.3. Headers

Key	Value
Authorization	Bearer {token}

10.7.4. Path Parameters

Parameter	Type
scheduled_job_id	string

10.8. Create a Scheduled Job

10.8.1. HTTP Request

```
POST https://gateway.flow.librestream.com/scheduledjobs/v1
```

10.8.2. Authorization

Bearer Token

Policy [Schedule_Admin](#) (on page 28)

10.8.3. Headers

Key	Value
Authorization	Bearer {token}

10.8.4. Request Body

10.8.4.1. JSON Representation

```
{
  "name": "API Scheduler Example Job 1",
  "description": "To Demo Scheduler Apis",
  "tasks":
  [
    {
      "taskType": "workflow", "
      "artifactId": "b62f2507-b4e7-4248-83d3-29c03a158fae"
    }
  ],
  "assignedUserIds": ["5d91f997ccfe6a0001dc9dc1"],
  "dueByDate": null,
  "priority": false
}
```

10.8.5. Response Body

10.8.5.1. JSON Representation

```
{
  "jobId": "615da9233ed8fe0021364451",
  "created": "2021-10-06T13:48:19.342407Z",
  "updated": "2021-10-06T13:48:19.342407Z",
  "state": "Scheduled",
  "name": "API Scheduler Example Job 1",
  "description": "To Demo Scheduler Apis",
  "priority": false,
  "dueByDate": null,
  "assignedUserIds": [
    "5d8cac54860b8100016b1987"],
  "tasks": [
    {
      "taskType": "Workflow",
      "taskId": "dbd38dd1-fc76-4a3a-8e4c-27a1e9a4148c",
      "artifactId": "b62f2507-b4e7-4218-83d3-29c03a158fae",
      "jobId": null,
      "clientJobId": null,
      "updated": null,
      "modified": null,
      "state": 0
    }
  ]
}
```

NB: artifactId is the workflowId.

11. TeamSettings

Team Settings can be managed via the APIs

Related information

[Set TeamSettings to Defaults \(on page 131\)](#)

[Set TeamSettings \(on page 131\)](#)

[Get TeamSetting \(on page 132\)](#)

[Get TeamSettings \(on page 133\)](#)

11.1. Set TeamSettings to Defaults

11.1.1. HTTP Request

```
POST https://gateway.flow.librestream.com/teamsettings/v1/defaults
```

11.1.2. Authorization

Bearer Token

Policy [TeamSettings_Admin \(on page 30\)](#)

11.1.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Type	application/json

11.2. Set TeamSettings

11.2.1. HTTP Request

```
POST https://gateway.flow.librestream.com/teamsettings/v1/
```

11.2.2. Authorization

Bearer Token

Policy [TeamSettings_Admin \(on page 30\)](#)

11.2.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Type	application/json

11.2.4. Request Body

11.2.4.1. JSON Representation

```
{
  "DisableJobRestart":true,
  "EnableJobNaming":false
}
```

11.3. Get TeamSetting

11.3.1. HTTP Request

```
GET https://gateway.flow.librestream.com/teamsettings/v1/{setting_key}
```

11.3.2. Authorization

Bearer Token

Policy [TeamSettings_Read](#) (on page 30)

11.3.3. Headers

Key	Value
Authorization	Bearer {token}

11.3.4. Path Parameters

Parameter	Type
setting_key (DisableJobRestart)	string

11.3.5. Response Body

11.3.5.1. JSON Representation

```
{
  true
}
```

11.4. Get TeamSettings

11.4.1. HTTP Request

```
GET https://gateway.flow.librestream.com/teamsettings/v1/
```

11.4.2. Authorization

Bearer Token

Policy [TeamSettings_Read](#) (on page 30)

11.4.3. Headers

Key	Value
Authorization	Bearer {token}

11.4.4. Response Body

11.4.4.1. JSON Representation

```
{
  "AutoDeviceSignOut":null,
  "AutoDownload":{
    "WorkflowUpdateMode":"Optional",
    "WorkflowUpdateLapsedTime":0
  },
  "CheckDeviceTime":false,
  "DefaultTimezone":"Etc/UTC",
  "DisableJobRestart":true,
  "DisableOldTemplates":false,
  "EnableJobNaming":false,
  "GoBackReason":"Optional",
  "PermanentDataEdit":false,
  "RemoteExpertSolution":null,
  "Reports":{
    "DefaultReport":null
  },
  "RequiresAbandonReason":false,
  "StoreDBOnExternalStorage":false,
  "SupportsOnsight":false,
  "WorkflowApproverMode":0
}
```

12. Trigger

Triggers can be managed via the APIs

Related information

[Get All Triggers \(on page 135\)](#)

[Get Trigger by Trigger Id \(on page 136\)](#)

[Create Trigger \(on page 137\)](#)

[Update Trigger \(on page 138\)](#)

[Archive Trigger \(on page 139\)](#)

[Unarchive Trigger \(on page 140\)](#)

12.1. Get All Triggers

12.1.1. HTTP Request

```
GET https://gateway.flow.librestream.com/trigger/v2
```

12.1.2. Authorization

Bearer Token

Policy [Trigger_Read \(on page 31\)](#)

12.1.3. Headers

Key	Value
Authorization	Bearer {token}

12.1.4. Response Body

12.1.4.1. JSON Representation

```
[
  {
    "triggerId": "61644119f2efe60001085778",
    "created": "2021-10-11T13:50:17Z",
    "updated": "2021-12-10T14:02:26Z",
    "name": "Trigger Api Ex 1",
    "description": null,
    "templateTags": [],
    "objectIds": [],
    "templateIds": [
      "a454a699-173f-429e-b3cd-fefd82d0a1e4"
    ],
    "triggerType": "Email",
    "triggerOptions": {
      "recipients": "",
      "sendCopyToUser": "true"
    },
    "reportTemplateId": "6163da95736e3c00014f9bee",
    "pdfRequirement": "FullImages",
    "disabled": false,
    "archived": false
  },
  {
    "triggerId": "5f02e48325b6dc00018b81e8",
    "created": "2020-07-06T08:44:51Z",
    "updated": "2020-07-06T08:44:51Z",
    "name": "Trigger Api Ex 2",
    "description": null,
    "templateTags": [],
    "objectIds": [],
    "templateIds": [],
    "triggerType": "Email",
    "triggerOptions": {},
    "reportTemplateId": null,
    "pdfRequirement": "NotRequired",
    "disabled": true,
    "archived": false
  }
]
```

12.2. Get Trigger by Trigger Id

12.2.1. HTTP Request

```
GET https://gateway.flow.librestream.com/trigger/v2/{triggerId}
```

12.2.2. Authorization

Bearer Token

12.2.3. Headers

Key	Value
Authorization	Bearer {token}

12.2.4. Path Parameters

Parameter	Type
triggerId	string

12.2.5. Response Body

12.2.5.1. JSON Representation

```
[
  {
    "triggerId": "61644119f2efe60001085778",
    "created": "2021-10-11T13:50:17Z",
    "updated": "2021-12-10T14:02:26Z",
    "name": "Trigger Api Ex 1",
    "description": null,
    "templateTags": [],
    "objectIds": [],
    "templateIds": [
      "a454a699-173f-429e-b3cd-fefd82d0a1e4"
    ],
    "triggerType": "Email",
    "triggerOptions": {
      "recipients": "",
      "sendCopyToUser": "true"
    },
    "reportTemplateId": "6163da95736e3c00014f9bee",
    "pdfRequirement": "FullImages",
    "disabled": false,
    "archived": false
  }
]
```

12.3. Create Trigger

12.3.1. HTTP Request

```
POST https://gateway.flow.librestream.com/trigger/v2
```

12.3.2. Authorization

Bearer Token

Policy [Trigger_Admin](#) (on page 30)

12.3.3. Headers

Key	Value
Authorization	Bearer {token}

12.3.4. Request Body

12.3.4.1. JSON Representation

```
{
  "disabled":true,
  "name":"API Trigger Ex 1",
  "triggerType":"Email"
}
```

12.3.5. Response Body

12.3.5.1. JSON Representation

```
{
  "triggerId": "61b70762ccc3a70q011c0462",
  "created": "2021-12-13T08:42:10Z",
  "updated": "2021-12-13T08:42:10Z",
  "name": "API Trigger Ex 2",
  "description": null,
  "templateTags": [],
  "objectIds": [],
  "templateIds": [],
  "triggerType": "Email",
  "triggerOptions": {},
  "reportTemplateId": null,
  "pdfRequirement": "NotRequired",
  "disabled": true,
  "archived": false
}
```

12.4. Update Trigger

12.4.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/trigger/v2/{triggerId}
```

12.4.2. Authorization

Bearer Token

Policy [Trigger_Admin](#) (on page 30)

12.4.3. Headers

Key	Value
Authorization	Bearer {token}

12.4.4. Path Parameters

Parameter	Type
triggerId	string

12.4.5. Request Body

12.4.5.1. JSON Representation

```
{
  "name": "APi Trigger Ex 1",
  "description": "Updating trigger description via api call",
  "templateTags": [],
  "objectIds": [],
  "templateIds": [],
  "triggerType": "Email",
  "triggerOptions": {"recipients": "", "sendCopyToUser": "true"},
  "pdfRequirement": "NotRequired",
  "reportTemplateId": null,
  "disabled": true
}
```

12.5. Archive Trigger

12.5.1. HTTP Request

```
DEL https://gateway.flow.librestream.com/trigger/v2/{triggerId}
```

12.5.2. Authorization

Bearer Token

Policy [Trigger_Admin \(on page 30\)](#)

12.5.3. Headers

Key	Value
Authorization	Bearer {token}

12.5.4. Path Parameters

Parameter	Type
triggerId	string

12.6. Unarchive Trigger

12.6.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/trigger/v2/unarchive/{triggerId}
```

12.6.2. Authorization

Bearer Token

Policy [Trigger_Admin \(on page 30\)](#)

12.6.3. Headers

Key	Value
Authorization	Bearer {token}

12.6.4. Path Parameters

Parameter	Type
triggerId	string

13. Users

Users can be managed via the APIs

Related information

[Get Users \(on page 141\)](#)

[Get User \(on page 142\)](#)

[Create User \(on page 143\)](#)

[Get User Permissions \(on page 144\)](#)

[Lock User \(on page 145\)](#)

[Reset User's Password \(on page 145\)](#)

[Unlock User \(on page 146\)](#)

[Update User Permissions \(on page 147\)](#)

[Delete User \(on page 148\)](#)

13.1. Get Users

13.1.1. HTTP Request

```
GET https://accounts.flow.librestream.com/api/team/user
```

13.1.2. Authorization

Bearer Token

Policy Any Valid Token

13.1.3. Headers

Key	Value
Authorization	Bearer {token}

13.1.4. Response Body

13.1.4.1. JSON Representation

```
[
  {
    "userId": "60df2148ad6cbd0001d86a1d",
    "username": "colin.pope@solarenergy.com",
    "emailAddress": "colin.pope@solarenergy.com",
    "name": "Colin Pope",
    "created": "2021-07-02T14:23:04.352Z",
    "locked": false
  },
  {
    "userId": "60df09e19108b2e627754f54",
    "username": "tegla.loroupe@solarenergy.com",
    "emailAddress": "tegla.loroupe@solarenergy.com",
    "name": "Tegla Loroupe",
    "created": "2021-06-01T15:14:52.000Z",
    "locked": false
  }
]
```

13.2. Get User

13.2.1. HTTP Request

```
GET https://accounts.flow.librestream.com/api/team/user/{user_id}
```

13.2.2. Authorization

Bearer Token

Policy Any Valid Token

13.2.3. Headers

Key	Value
Authorization	Bearer {token}

13.2.4. Path Parameters

Parameter	Type
userId	string

13.2.5. Response Body

13.2.5.1. JSON Representation

```
{
  "userId": "60df2148ad6cbd0001d86a1d",
  "username": "colin.pope@solarenergy.com",
  "emailAddress": "colin.pope@solarenergy.com",
  "name": "Colin Pope",
  "created": "2021-07-02T14:23:04.352Z",
  "locked": false
}
```

13.3. Create User

13.3.1. HTTP Request

POST https://accounts.flow.librestream.com/api/team/user

13.3.2. Authorization

Bearer Token

Policy [Trigger_Admin \(on page 30\)](#)

13.3.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Type	application/json

13.3.4. Request Body

13.3.4.1. JSON Representation

```
{
  "username": "chege.naengop@solarenergy.com",
  "emailAddress": "chege.naengop@solarenergy.com",
  "name": "Chege Naengop"
}
```

13.3.5. Response Body

13.3.5.1. JSON Representation

```
{
  "password": "3DfJ/84dY",
  "teamName": "solarenergy",
  "mode": 0,
  "userId": "60e3054c04cb64000128d658",
  "username": "chege.naengop@solarenergy.com",
  "emailAddress": "chege.naengop@solarenergy.com",
  "name": "Chege Naengop",
  "created": "2021-07-05T13:12:44.7442233Z",
  "locked": false
}
```

13.4. Get User Permissions

13.4.1. HTTP Request

```
GET https://accounts.flow.librestream.com/api/team/user/{user_id}/permissions
```

13.4.2. Authorization

Bearer Token

Policy [Trigger_Admin](#) (on page 30)

13.4.3. Headers

Key	Value
Authorization	Bearer {token}

13.4.4. Path Parameters

Parameter	Type
user_id	string

13.4.5. Response Body

13.4.5.1. JSON Representation

```
{
  "dataViewer": true,
  "executor": true
}
```

13.5. Lock User

13.5.1. HTTP Request

```
POST https://accounts.flow.librestream.com/api/team/{user_id}/lock
```

13.5.2. Authorization

Bearer Token

Policy [Trigger_Admin \(on page 30\)](#)

13.5.3. Headers

Key	Value
Authorization	Bearer {token}

13.5.4. Path Parameters

Parameter	Type
user_id	string

13.6. Reset User's Password

13.6.1. HTTP Request

```
POST https://accounts.flow.librestream.com/api/team/{user_id}/resetpassword
```

13.6.2. Authorization

Bearer Token

Policy [Trigger_Admin \(on page 30\)](#)

13.6.3. Headers

Key	Value
Authorization	Bearer {token}

13.6.4. Path Parameters

Parameter	Type
user_id	string

13.6.5. Response Body

13.6.5.1. JSON Representation

```
{  
  "password": "3DfJ/84dY"  
}
```

13.7. Unlock User

13.7.1. HTTP Request

```
POST https://accounts.flow.librestream.com/api/team/{user_id}/unlock
```

13.7.2. Authorization

Bearer Token

Policy [Trigger_Admin](#) (on page 30)

13.7.3. Headers

Key	Value
Authorization	Bearer {token}

13.7.4. Path Parameters

Parameter	Type
user_id	string

13.8. Update User Details

13.8.1. HTTP Request

```
PUT https://accounts.flow.librestream.com/api/team/user/{user_id}
```

13.8.2. Authorization

Bearer Token

Policy [Trigger_Admin \(on page 30\)](#)

13.8.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Type	application/json

13.8.4. Path Parameters

Parameter	Type
user_id	string

13.8.5. Request Body

13.8.5.1. JSON Representation

```
{
  "Name": "Chege Naengop"
}
```

13.9. Update User Permissions

13.9.1. HTTP Request

```
PUT https://accounts.flow.librestream.com/api/team/user/{userId}/permissions
```

13.9.2. Authorization

Bearer Token

Policy [Trigger_Admin \(on page 30\)](#)

13.9.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Type	application/json

13.9.4. Path Parameters

Parameter	Type
userId	string

13.9.5. Request Body

13.9.5.1. JSON Representation

```
{
  "admin": False,
  "dataViewer": True,
  "executor": True
}
```

13.10. Delete User

13.10.1. HTTP Request

```
DEL https://accounts.flow.librestream.com/api/team/user/{user_id}
```

13.10.2. Authorization

Bearer Token

Policy [Trigger_Admin](#) (on page 30)

13.10.3. Headers

Key	Value
Authorization	Bearer {token}

13.10.4. Path Parameters

Parameter	Type
user_id	string

14. Workflows

Workflows can be managed via the APIs

Related information

[Get Workflows \(on page 151\)](#)

[Get Workflow \(on page 152\)](#)

[Update Workflow Metadata \(on page 153\)](#)

[Upload a Workflow \(on page 154\)](#)

[Set Active Version \(on page 159\)](#)

[Delete Workflow Version \(on page 160\)](#)

[Delete Workflow \(on page 160\)](#)

14.1. Get Workflows

14.1.1. HTTP Request

```
GET https://gateway.flow.librestream.com/workflows/v2
```

14.1.2. Authorization

Bearer Token

Policy [Workflow_Read \(on page 33\)](#)

14.1.3. Headers

Key	Value
Authorization	Bearer {token}

14.1.4. Response Body

14.1.4.1. JSON Representation

```
[
  {
    "workflowId": "75b81832-257b-4a7a-9e05-g121acc11a67",
    "created": "2020-01-14T11:53:47Z",
    "lastUpdated": "2020-02-04T08:18:49Z",
    "teamName": "solarenergy",
    "activeVersionId": "873b6fc9-6e62-4d8f-aac5-42c3aa8235a8",
    "name": "Panel Repair Workflow",
    "description": "",
    "isArchived": false,
    "liveObjectIds": [],
    "versions": [
      {
        "versionId": "48254f9a-80ab-4b09-b041-0671716d73d3",
        "fileReference": "5e1da94f1ab7e300017ab586",
        "fileSize": 159483693,
        "uploaded": "2020-01-14T11:53:47Z",
        "authorId": "5da480b75ed0570001d197e2",
        "authorName": "shankar.lakshman@solarenergy.com",
        "version": 1,
        "isArchived": false
      },
      {
        "versionId": "873b6fc9-6e62-4d8f-aac5-42c3aa8235a8",
        "fileReference": "5e1dad09b4ccd1000102e8cf",
        "fileSize": 159483693,
        "uploaded": "2020-01-14T12:04:22Z",
        "authorId": "5da480b75ed0570001d197e2",
        "authorName": "shankar.lakshman@solarenergy.com",
        "version": 2,
        "isArchived": false
      }
    ]
  }
]
```

14.2. Get Workflow

14.2.1. HTTP Request

POST <https://gateway.flow.librestream.com/workflows/v2/{workflowId}>

14.2.2. Authorization

Bearer Token

Policy [Workflow_Read \(on page 33\)](#)

14.2.3. Headers

Key	Value
Authorization	Bearer {token}

14.2.4. Path Parameters

Parameter	Type
workflowId	string

14.2.5. Response Body

14.2.5.1. JSON Representation

```
{
  "workflowId": "75b81832-257b-4a7a-9e05-g121acc11a67",
  "created": "2020-01-14T11:53:47Z",
  "lastUpdated": "2020-02-04T08:18:49Z",
  "teamName": "solarenergy",
  "activeVersionId": "873b6fc9-6e62-4d8f-aac5-42c3aa8235a8",
  "name": "Panel Repair Workflow",
  "description": "",
  "isArchived": false,
  "liveObjectIds": [],
  "versions": [
    {
      "versionId": "48254f9a-80ab-4b09-b041-0671716d73d3",
      "fileReference": "5e1da94f1ab7e300017ab586",
      "fileSize": 159483693,
      "uploaded": "2020-01-14T11:53:47Z",
      "authorId": "5da480b75ed0570001d197e2",
      "authorName": "shankar.lakshman@solarenergy.com",
      "version": 1,
      "isArchived": false
    },
    {
      "versionId": "873b6fc9-6e62-4d8f-aac5-42c3aa8235a8",
      "fileReference": "5e1dad09b4ccd1000102e8cf",
      "fileSize": 159483693,
      "uploaded": "2020-01-14T12:04:22Z",
      "authorId": "5da480b75ed0570001d197e2",
      "authorName": "shankar.lakshman@solarenergy.com",
      "version": 2,
      "isArchived": false
    }
  ]
}
```

14.3. Update Workflow Metadata

14.3.1. HTTP Request

```
PUT https://gateway.flow.librestream.com/workflows/v2/{workflowId}
```

14.3.2. Authorization

Bearer Token

Policy [Workflow_Modify \(on page 33\)](#)

14.3.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Type	application/json

14.3.4. Path Parameters

Parameter	Type
workflowId	string

14.3.5. Request Body

14.3.5.1. JSON Representation

```
{
  "name": "Panel Repair Workflow - AP-6 Type",
  "description": "Ensure this workflow is used on the latest AP-6 type panels",
  "tags": ["Panel", "Repair", "AP-6"],
  "liveObjectIds": ["5e4172bc9a21690001f361f1", "5e46a021785ccf0001e9a2f6"]
}
```

N.B. By default workflows are available for use for any user with the required permissions, however it is possible to restrict the usage of workflow to specified individual users or user groups. The *liveObjectIds* parameter is used to specify the users or groups; for a user include the *userId* and for a group include the *groupId*.

14.4. Upload a Workflow

14.4.1. Overview

As Workflow files have the potential to be quite large; primarily when they include large assets such as videos, images and files, The Workflow Upload API provides a way to reliably upload those files by blocking them into a sequence of parts that can be uploaded individually.

By using this API the application uploads a file in part, allowing it to recover from a failed request more reliably. It means an application only needs to retry the upload of a single part instead of the entire file.

An additional benefit of blocked uploads is that parts can be uploaded in parallel, allowing for a potential performance improvement.

The process for uploading a Workflow requires a sequence of API calls to be made

1. [A Workflow UploadSession is Created \(on page 156\)](#): This session is used as a reference for the later calls and sets expectations for the block sizes and files sizes.
2. [File blocks are Uploaded \(on page 157\)](#): The file is uploaded as a series of blocks (N.B. if the file is small enough it can be uploaded as a single block).
3. [The Workflow Upload Session is Completed \(on page 158\)](#): This call is made once all blocks have successfully been uploaded, it closes the session and defines metadata on the Workflow.

14.4.2. Example

I have a file of size 150 KB which I will uploaded in blocks of 100 KB, I multiply both of these by 1024 to calculate the file size and block size

```
BLOCKSIZE = 102400  
FILESIZE = 153600
```

I make an initial call to create the workflow upload session

```
curl -XPOST  
-H 'Authorization: Bearer eyJhbGciOiJSUzI1NiIsImtpZCI6...'  
-H "Content-type: application/json"  
-d '{"blockSize": 102400, "fileSize": 153600, "friendlyFileName": "Workflow1.zip"}'  
'https://gateway.flow.librestream.com/workflows/v2/upload'
```

As a response I receive an Id for the upload session

```
{  
  "id": "60578b426a9eb4000114ec13"  
}
```

I then make two separate calls for each block, each referencing the upload session Id in the Url. The first uploads the first 100 KB and the second the remaining 50 KB (N.B. the order of upload is not important)

14.4.2.1. Block 1

```
curl -XPOST
-H 'Authorization: Bearer eyJhbGciOiJSUzI1NiIsImtpZCI6...'
-H 'Content-Length: 102400'
-H 'Content-Range: bytes 0-102399/153600'
-d 'PK\x03\x04\x14\x00\x00\x00\x08\x00...'
'https://gateway.flow.librestream.com/workflows/v2/upload/60578b426a9eb4000114ec13'
```

14.4.2.2. Block 2

```
curl -XPOST
-H 'Authorization: Bearer eyJhbGciOiJSUzI1NiIsImtpZCI6...'
-H 'Content-Length: 102400'
-H 'Content-Range: bytes 102400-153599/153600'
-d '\xf7\xfb\xb7\x05\xea~\x01\xdf...'
'https://gateway.flow.librestream.com/workflows/v2/upload/60578b426a9eb4000114ec13'
```

Once those are both completed I then complete the upload session by defining the metadata for the workflow

```
curl -XPOST
-H 'Authorization: Bearer eyJhbGciOiJSUzI1NiIsImtpZCI6...'
-H "Content-type: application/json"
-d '{ "uploadId": "60578b426a9eb4000114ec13", "name": "My First Workflow" }'
'https://gateway.flow.librestream.com/workflows/v2'
```

Related information

[Create Workflow Upload Session \(on page 156\)](#)

[Upload Workflow Block \(on page 157\)](#)

[Finalize Workflow Upload \(on page 158\)](#)

14.4.3. Create Workflow Upload Session

14.4.3.1. HTTP Request

```
POST https://gateway.flow.librestream.com/workflows/v2/upload
```

14.4.3.2. Authorization

Bearer Token

Policy [Workflow_Create \(on page 32\)](#)

14.4.3.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Type	application/json

14.4.3.4. Request Body

14.4.3.4.1. JSON Representation

```
{
  "blockSize": {BLOCKSIZE},
  "fileSize": {FILESIZE},
  "friendlyFileName": "Workflow1.zip"
}
```

14.4.3.5. Response Body

14.4.3.5.1. JSON Representation

```
{
  "id": "60578b426a9eb4000114ec13"
}
```

14.4.4. Upload Workflow Block

14.4.4.1. HTTP Request

```
POST https://gateway.flow.librestream.com/workflows/v2/upload/{uploadId}
```

14.4.4.2. Authorization

Bearer Token

Policy [Workflow_Create](#) (on page 32)

14.4.4.3. Headers

Key	Value
Authorization	Bearer {token}
Content-Length	{BLOCKSIZE}
Content-Range	bytes {i x BLOCKSIZE}-{(i + 1) x BLOCKSIZE}/{FILESIZE}

14.4.4.4. Request Body

```
{block_bytes}
```

14.4.4.4.1. If the block was the final block and all blocks been uploaded a 204 is returned

14.4.4.4.2. If the block was not the final block but has been uploaded a 202 is returned

14.4.4.5. Notes

- Workflow file size limit is 200 MB
- File chunk size limit is 100 KB

14.4.5. Finalize Wordflow Upload

14.4.5.1. HTTP Request

```
POST https://gateway.flow.librestream.com/workflows/v2
```

14.4.5.2. Authorization

Bearer Token

Policy [Workflow_Create](#) (on page 32)

14.4.5.3. Headers

Key	Value
Authorization	Bearer {token}

14.4.5.4. Request Body

14.4.5.4.1. JSON Representation

```
{
  "uploadId": "60578b426a9eb4000114ec13",
  "name": "Repair Process",
  "description": "Repair Process for all new and current Plant Machinery",
  "tags": ["Repairs", "Level 1 Process"],
  "versionNotes": "First version of the process"
}
```

14.4.5.5. Response Body

14.4.5.5.1. JSON Representation

```
{
  "workflowId": "4dcb1a1b-782f-4167-8a87-ca6b31ae877d"
}
```

14.5. Set Active Version

14.5.1. HTTP Request

POST https://gateway.flow.librestream.com/workflows/v2/{workflowVersionId}/activate

14.5.2. Authorization

Bearer Token

Policy [Workflow_Modify \(on page 33\)](#)

14.5.3. Headers

Key	Value
Authorization	Bearer {token}

14.5.4. Path Parameters

Parameter	Type
workflowVersionId	string

14.6. Delete Workflow Version

14.6.1. HTTP Request

```
DELETE
https://gateway.flow.librestream.com/workflows/v2/{workflowId}/{workflowVersionId}
```

14.6.2. Authorization

Bearer Token

Policy [Workflow_Delete](#) (on page 32)

14.6.3. Headers

Key	Value
Authorization	Bearer {token}

14.6.4. Path Parameters

Parameter	Type
workflowId	string
workflowVersionId	string

14.7. Delete Workflow

14.7.1. HTTP Request

```
DELETE https://gateway.flow.librestream.com/workflows/v2/{workflowId}
```

14.7.2. Authorization

Bearer Token

Policy [Workflow_Delete](#) (on page 32)

14.7.3. Headers

Key	Value
Authorization	Bearer {token}

14.7.4. Path Parameters

Parameter	Type
workflowId	string