Guide for software integration with automation API 5.0

May 2023

Document Version: 5.0

Contents

Introduction 4

About this document 4

Overview 5

Accounts return (AR) submission process 5

Timelines for the live online form 6

Draft trust financial statements process 6

Registering your software product 7

Annual update of client secret ID’s 7

Getting an OAuth 2.0 access token 10

Step 1: Invoke the Authorisation endpoint 10

Step 2: Receive authorisation results 13

Step 3: Exchange authorisation code for access token 13

Step 4: Refreshing an access token 15

Step 5: Calling the Financial Return Submission API 16

API specification for financial return submission 17

Draft trust financial statements 17

Data requirements for submission 19

Data validations 21

Data set information 21

API version 22

API endpoints 22

Create or update Submission 22

Delete Submission (available for testing purposes only) 27

Get submission (available for testing purposes only) 28

Key environment variables 31

Supplier test 31

User acceptance testing (UAT) 32

Chart of accounts (CoA) 33

CoA version 33

CoA account code structure 34

Glossary 36

HTTP status codes 38

API error codes 39

AR202/23 release changes 40

Document control 41

Appendix 44

Introduction

Since 2020 the Department for Education (DfE) has improved end-to-end financial reporting for academy trusts (trusts). As part of this work, the DfE has developed an application programming interface (API) to support the automated submission of a trusts trial balance data via their financial management system (FMS). This automated data submission is based on a unified chart of accounts structure which academy trusts must use but can either adopt or map to.

The data submitted via the API will be used to pre-populate the academies account return which trusts are required to submit to the DfE. This will deliver benefits across the sector by saving time and costs of submitting financial data.

About this document

The document will serve as a helpful guide for software suppliers to help prepare for the integration with the API and applies to the API that is to be used with AR22/23 collection.

The document contains details relating to both the consent process and the financial return submission API itself.

This document is intended for use by business analysts, technical architects, and software developers from software supplier organisations. To find out more about the academy accounts return online form use this link.

For any queries, you can email api.finance@education.gov.uk .

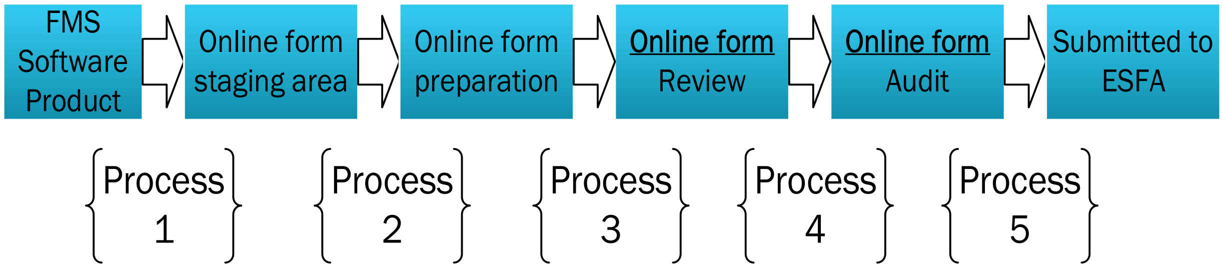
Overview

This API is designed to support academy trusts with completion of financial returns to DfE. This is normally done using an online form allowing the trust user to fill in the details and get immediate feedback of any issues. By using this API, the trust can transfer much of the information directly from their finance software saving time and helping to ensure better accuracy.

Accounts return (AR) submission process

The following diagram shows the different stages in the completion and submission of the AR form:

Stages for accounts return online form



- **Process 1**: API used here to transfer data to the online form staging area. There is only ever one set of data stored and it can be updated/overwritten
- **Process 2**: The trust preparer will need to use the form to select data from an API to be used to prepopulate the AR form.
- **Process 3:** Once the trust user has completed the whole AR form they will submit for review.
- **Process 4**: Once the internal review is successful the trust can submit the return for external review by auditors.
- **Process 5:** Only once the auditor has approved the AR form can the data be submitted to ESFA.

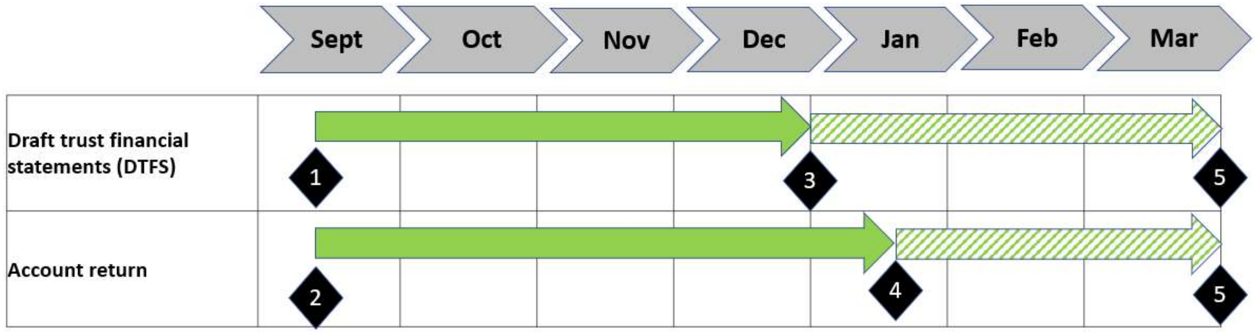
The API described within this document is used on the first stage allowing trusts to transfer a set of data across to the online form staging area. Once received, the trust preparer can choose whether to use the data to pre-populate the online form.

Note: The draft trust financial statements functionality is scheduled to go live on 5 September 2023. If trusts are submitting data for the draft trust financial statements prior to the AR's scheduled go live date of 12 September 2022, there will be no facility to submit data into the AR form.

Timelines for the live online form

The planned timelines for the availability of the ESFA automated financial on-line form 2022/23 are as follows:

Automated ESFA financial on-line form availability AR 2022/23



1 **12th September 2023** - AR open for trusts to run automated DTFS only

2 **12th September 2023** - AR open for trusts to run DTFS and complete their AR (via automation or manual input)

3 **31st December 2023** - Financial statements statutory deadline for submission (not via AR online form), live DTFS functionality remains available until late March / early April 2023

4 **30th January 2024** - AR statutory deadline for submission, form remains open for input

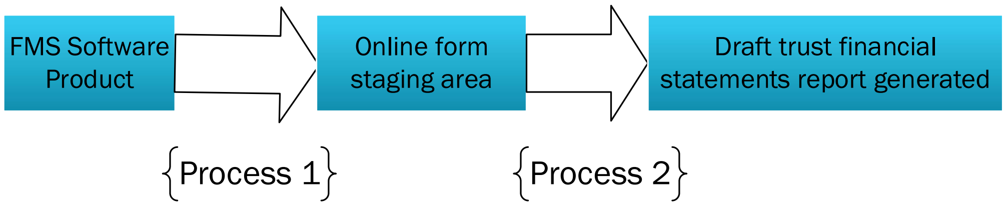
5 **March 2024 -** live AR online form closed to run DTFS and AR input

Draft trust financial statements process

The draft trust financial statements report is scheduled to be available from 12th September 2023 via the AR on-line form. With exception of the draft trust financial statements report, the full AR form will also be available for input from this date..

The following diagram shows the different stages in the generation of the draft trust financial statements report:

Stages for generating draft trust financial statements



- **Process 1**: API used here to transfer data to the online form staging area. There is only ever one set of data stored and it can be updated/overwritten
- **Process 2**: From within the online form, the trust preparer clicks on the ‘Draft trust financial statements’ link to populate the draft trust financial statements report with the latest API data set transferred

Registering your software product

Before calling the API, a software supplier will need to register their software with DfE. This is an important step in the standard authorisation process and allows us to issue unique codes to that software product. The following information will need to be provided to the DfE for each product to be registered:

1. Name of the product (which should be unique for that software organisation).
2. One or more redirect URLs which must be full URL strings. These are pre-registered addresses to which an authorisation code is returned. Since our service will only accept pre-registered URLs, it will be necessary to define all redirect URLs for all environments for that particular software product. For more details please refer to standard OAuth 2 documentation. Note that only https is supported.

Once the software product is registered, DfE will provide *client_id*, *client_secret and subscription_key* which you will need to use as part of the subsequent process.

The DfE has introduced a self-service portal (developer hub) to register your software product and subscribe to the submission API. A software supplier can register the application and can generate the client credentials and also the subscription keys for the API. The user can add the redirect URLs to the application at any point of time.

Link to the DfE developer hub - https://dfe-developerhub.education.gov.uk/

It is important to note that while software products will be pre-registered with the DfE, this will not include any assurance review. Therefore, registration does not act as offering an official approval for a particular software product.

Note: The suppliers can register the product using portal https://dfe-developerhub.education.gov.uk/ and it will go through the approval process. The approval for each environment can take up to 5 working days.

For any queries, you can email at api.finance@education.gov.uk

Annual update of client secret ID's

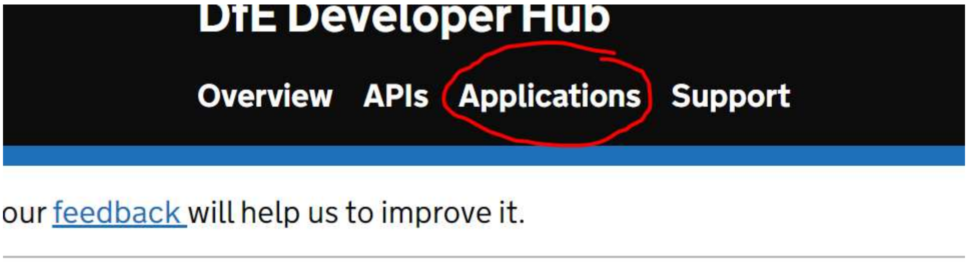
For security reasons, the client secret used in the API will expire 2 years from the time it was last generated.

We recommend suppliers update their subscription keys annually as part of their 'business as usual' processes, which will also ensure you are using the most up to date contact details. This is a really quick and easy process, and guidance is provided below. Please note that this is an automated procedure and no approval process is involved.

If you have questions or issues with this process, please contact API.Finance@education.gov.uk

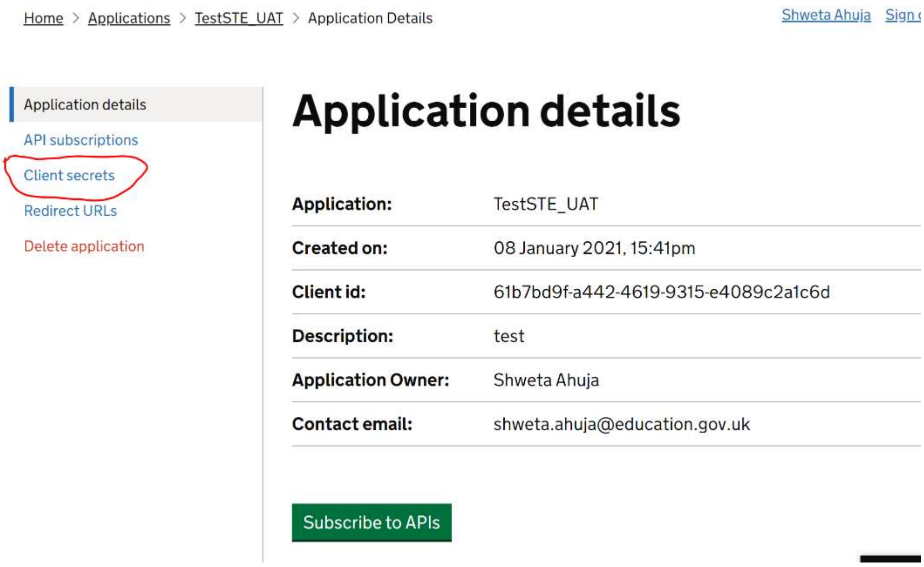
How to regenerate your client secret ID

1. Log into the developer hub using your existing account: https://dfe-developerhub.education.gov.uk/
2. Once logged in, click on ‘Application’



our feedback will help us to improve it.

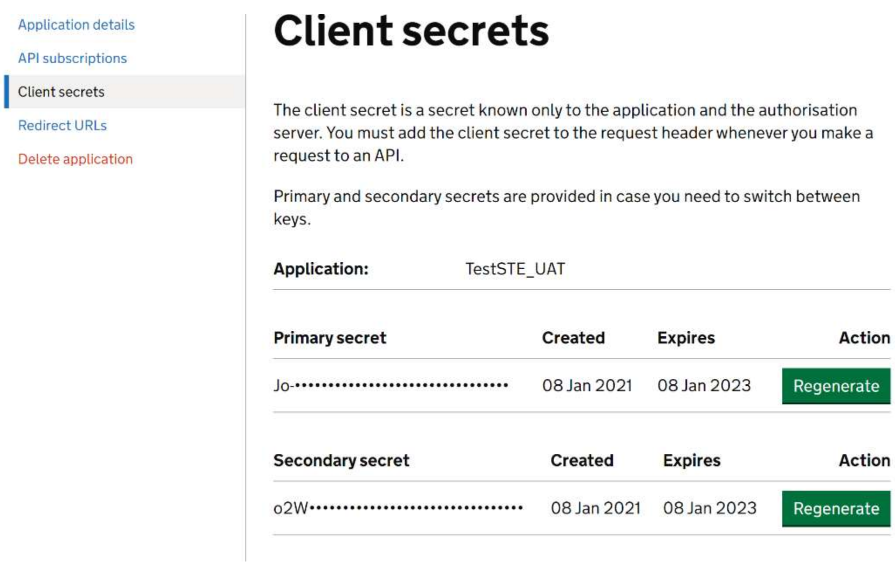
3. In the left-hand menu, click on ‘Client secret’



Application details

Application:	TestSTE_UAT
Created on:	08 January 2021, 15:41pm
Client id:	61b7bd9f-a442-4619-9315-e4089c2a1c6d
Description:	test
Application Owner:	Shweta Ahuja
Contact email:	shweta.ahuja@education.gov.uk

4. Under the ‘Primary secret’ heading click on the ‘Regenerate’ button



Client secrets

The client secret is a secret known only to the application and the authorisation server. You must add the client secret to the request header whenever you make a request to an API.

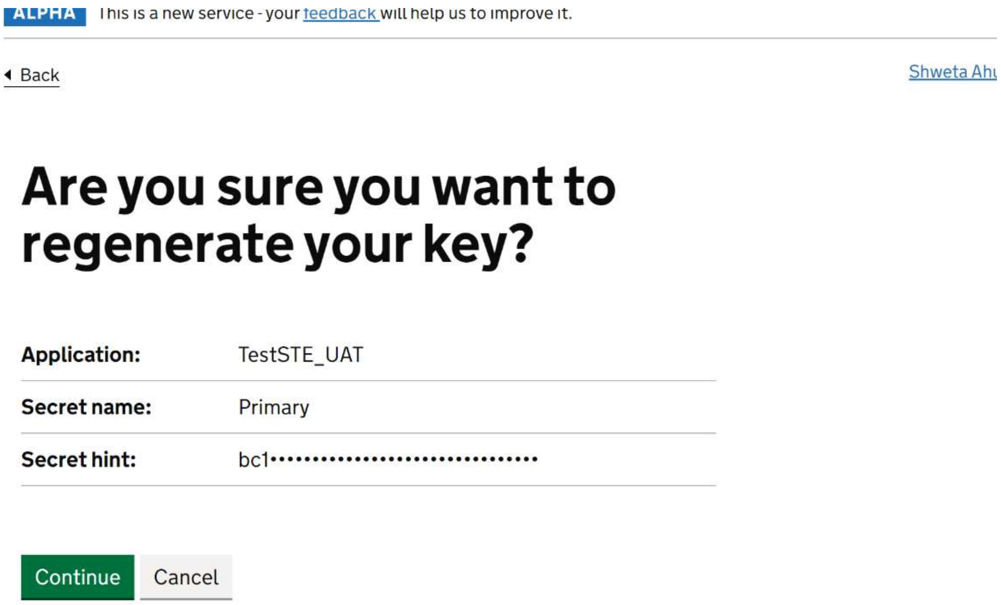
Primary and secondary secrets are provided in case you need to switch between keys.

Application: TestSTE_UAT

Primary secret	Created	Expires	Action
Jo-••••••••••••••••••••••••••••••••	08 Jan 2021	08 Jan 2023	Regenerate

Secondary secret	Created	Expires	Action
o2W••••••••••••••••••••••••••••••••	08 Jan 2021	08 Jan 2023	Regenerate

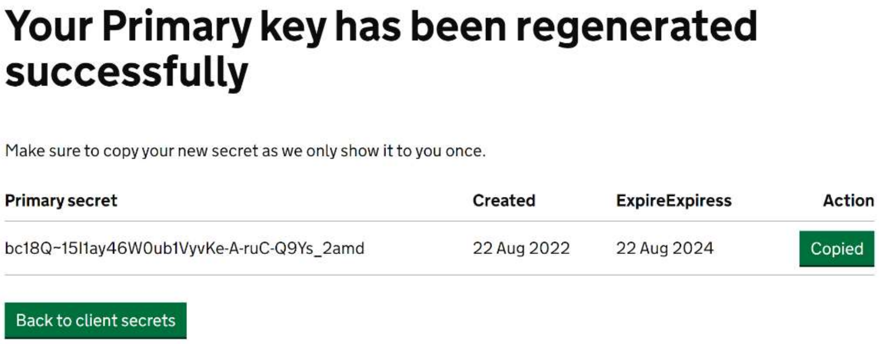
5. You will then be presented with the screen below, please click on the ‘Continue’ button


ALPHA
This is a new service - your feedback will help us to improve it.
Back
Shweta Ahu

Are you sure you want to regenerate your key?

Application:	TestSTE_UAT
Secret name:	Primary
Secret hint:	bc1••••••••••••••••••••••••••••••••

6. You will then receive a message to say ‘Your primary key has been regenerated successfully’



Your Primary key has been regenerated successfully

Make sure to copy your new secret as we only show it to you once.

Primary secret	Created	ExpireExpiress	Action
bc18Q~15l1ay46W0ub1VyvKe-A-ruC-Q9Ys_2amd	22 Aug 2022	22 Aug 2024	Copied

Getting an OAuth 2.0 access token

It is important to recognise that trusts retain ownership of this data so as a pre-requisite to calling the API, we ask that trusts approve the transfer of this data. The end user provides consent to connect with the DfE without sharing their access credentials. The end user who intends to provide that consent must have the specific role of ‘Data Transfer Approver’ (DTA) assigned in the DfE’s ‘IDAMS’ identity management service.

We use the open standard OAuth 2.0 for API authorisation. Suppliers will need to implement the authorisation user journey listed below to receive the OAuth 2.0 access token.

! You must not modify this journey in any way.

Step 1: Invoke the Authorisation endpoint

You must call the authorise endpoint to invoke the consent approval process. Please note that you need to provide either UPIN or company number below, and not both.

```
GET https://{authorize_base_url}
&response_type=code
&client_id={client_id}
&redirect_uri={redirect_url}
&nonce=defaultNonce
&scope=openid offline_access {fms_api_scope_url}
&prompt=login
&upin={trust_upin}
&companyNumber={companyhouse_number}
&apifamily={apiFamily}
&state={state}
```

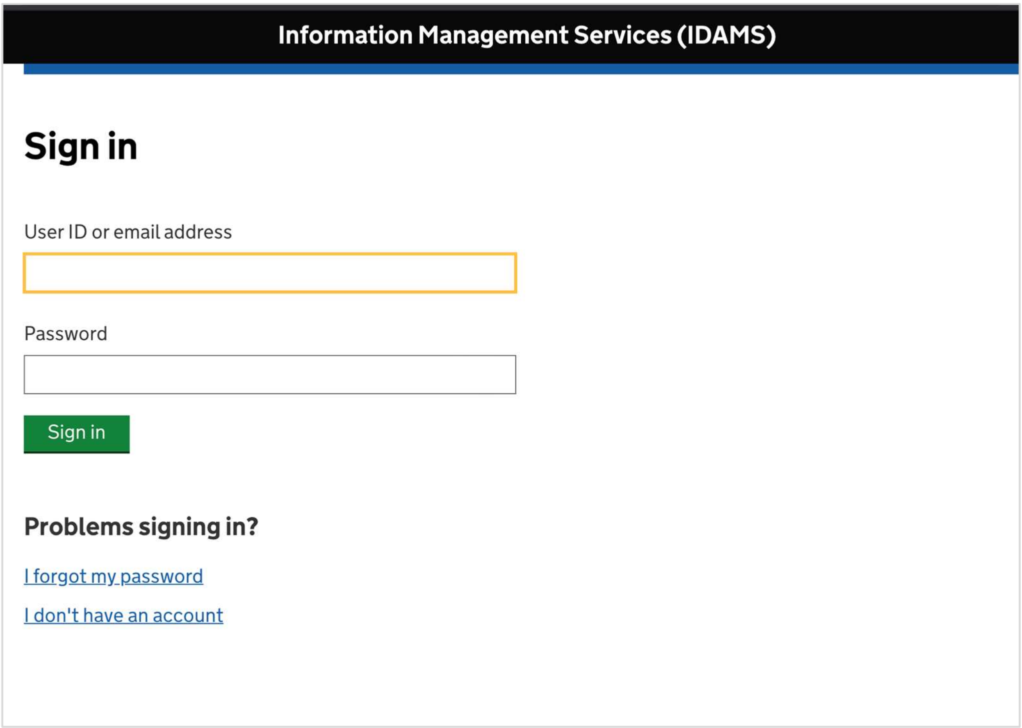
An example of the request to authorise endpoint is below:

```
https://dfeb2cpreprod.b2clogin.com/dfeb2cpreprod.onmicrosoft.com/oauth2
/v2.0/authorize?p=b2c_1a_consent_dfe&client_id=740c9a67-066c-4c10-a1c5-
63bc7c7844b5&nonce=defaultNonce&redirect_uri=https%3A%2F%2Fjwt.ms%2F&sc
ope= openid%20offline_access%20
https%3A%2F%2Fdfeb2cpreprod.onmicrosoft.com%2Fbfrp-
ar%2Fapi.submit&response_type=code&prompt=login&upin=770015&apiFamily=b
frp-ar&state=browser
```

! Do not include your client secret in this request.

Parameter	Description
response_type	The OAuth 2.0 response type. Currently the only acceptable value is `code` .
client_id	The client id associated with the FMS supplier application. Please note that the client id is specific to the DfE Azure B2C environment and is different from any client id provided as part of integrated testing with DfE Sign In.
redirect_uri	The HTTPS URL that we use to send the authorisation results back to your application after successful (or unsuccessful) authorisation. The URL needs to be encoded e.g. https%3A%2F%2Fjwt.ms This must match one of the redirect URLs specified during the application registration process.
scope	A space-delimited list of scopes you would like to have permission to access on behalf of your user. Must be percent-encoded, so spaces must be represented as either %20 or +. `openid offline_access {fms_api_scope_url}` E.g. in supplier test, {fms_api_scope_url}= https://dfeb2cpreprod.onmicrosoft.com/bfrp-ar/api.submit. The URL encoded scope would look like `openid%20offline_access%20` `https%3A%2F%2Fdfeb2cpreprod.onmicrosoft.com%2Fbfrp-ar%2Fapi.submit`
Nonce	A value to bind token to the client. The expected value is `defaultNonce`
prompt	Specifies whether the authorisation server prompts the End-User for consent. Currently the only acceptable value is `login`.
upin	Unique identifier for an academy trust. Must be six digits.
companynumber	Company House Number for an academy trust. Must be eight digits. **! Please provide either UPIN or Company House Number as providing both will lead to error**
apifamily	Currently the only valid value is `bfrp-ar`
State (optional)	An opaque value used to maintain state between the request and callback and to prevent tampering as described in the OAuth 2.0 specification. This is passed back to your application via the redirect_url.

This end point will open a HTML page where the user will be asked to login to the DfE's identity management service 'Pirean Access 1' or 'IDAMS'. Having logged in, they will then be presented with a consent page asking for approval to the transfer of data via the API. The user who gives consent will need to have been assigned the specific role of 'Data Transfer Approver' in the IDAMS system, this would normally be done by the Trusts' super-user.



Information Management Services (IDAMS)

Sign in

User ID or email address

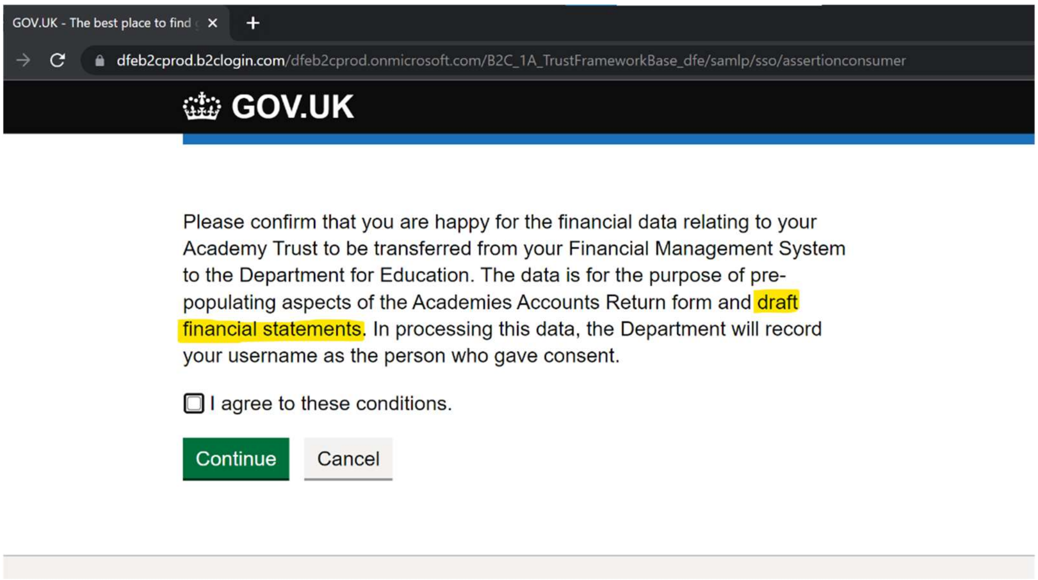
Password

Problems signing in?

I forgot my password

I don't have an account

The user needs to login and give the consent to transfer the data. The message was updated to also add consent to populate the draft trust financial statements.



GOV.UK - The best place to find
dfeb2cprod.b2clogin.com/dfeb2cprod.onmicrosoft.com/B2C_1A_TrustFrameworkBase_dfe/samlp/sso/assertionconsumer
GOV.UK

Please confirm that you are happy for the financial data relating to your Academy Trust to be transferred from your Financial Management System to the Department for Education. The data is for the purpose of pre-populating aspects of the Academies Accounts Return form and draft financial statements. In processing this data, the Department will record your username as the person who gave consent.

☐ I agree to these conditions.

Step 2: Receive authorisation results

You need to create an endpoint in your application to receive the authorisation results, which needs to support an HTTP GET to the redirect URL you specified.

We'll redirect the user's browser back to your endpoint once the user has granted your application the requested authority.

This authorisation code is valid for 10 mins.

Example of the authorisation response:

```
GET https://{redirect_url}?code={authorisation_code}
```

In case of error, you will receive this response:

GET **Error! Hyperlink reference not valid.** description}

Please refer to OAuth2.0 error responses for the list of errors. You will need to incorporate appropriate error handling to display user friendly error messages to the end users.

Step 3: Exchange authorisation code for access token

When you receive the authorisation code, you must exchange this for an access token **within 10 mins** before it expires.

Do this via a POST to our token endpoint.

Include the request parameters in the request body as a URL encoded string, not as request headers.

Example of a request to exchange authorisation code

```
POST https://{token_base_url}
Request body:
Content-Type application/x-www-form-urlencoded
grant_type authorization_code
code {authorisation code}

scope openid offline_access {fms_api_scope_url}

client_id {client_id}

client_secret {client_secret}
```

Parameter	Description
grant_type	The OAuth 2.0 grant type. Currently the only acceptable value is authorization_code
code	The authorisation code you received from us in the previous step.
client_id	The client id associated with the FMS supplier application.
client_secret	The client secret associated with the FMS supplier application
Scope	openid offline_access {fms_api_scope_url} E.g. in supplier test, {fms_api_scope_url}= https://dfeb2cpreprod.onmicrosoft.com/bfrp-ar/api.submit. The scope would look like openid offline_access https://dfeb2cpreprod.onmicrosoft.com/bfrp-ar/api.submit

Note: for suppliers who were involved in the private beta release of this API in November 2019, you will need a new client_id and client_secret, with new endpoints from Azure B2C.

The response contains the access token used for calling the APIs and a refresh token used to obtain a new access token once the current one expires.

Access_token and id_token will expire after 10 minutes.

Example of a successful exchange response:

```
{
  "access_token":"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtrZ..."
  “expires_in”:”600,
  "id_token":"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZ...",
  "token_type":"Bearer",
  "id_token_expires_in":600,
  "scope":"openid offline_access",
  "refresh_token":"MmZmNTMzMGYtZGRhYi00MDI1LWFiNWUtZjc2...",
  "refresh_token_expires_in":3600,
}
```

Step 4: Refreshing an access token

When the access token expires, you will need to send a POST request to exchange a refresh token for a new access token. The response will include a new refresh token that will replace the previous one and must be retained and used the next time the access token expires.

Refresh tokens will expire after 60 minutes.

You should send the POST request to:

```
POST https://{token_base_url}
Request body:
Content-Type application/x-www-form-urlencoded
grant_type refresh_token
refresh_token {refresh_token}

client_id {client_id}

client_secret {client_secret}

scope openid offline_access {fms_api_scope_url}
```

Example of a successful token response

```
{
  "access_token":"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtrZ..."
  “expires_in”:”600,
  "id_token":"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZ...",
  "token_type":"Bearer",
  "id_token_expires_in":600,
  "scope":"openid offline_access",
  "refresh_token":"MmZmNTMzMGYtZGRhYi00MDI1LWFiNWUtZjc2...",
  "refresh_token_expires_in":3600,
}
```

! You must store your access and refresh tokens securely and not share them with anyone else.

Step 5: Calling the Financial Return Submission API

You will need to use the access token and subscription key to call the Financial Return Submission API

Example of a call to a user-restricted API

```
PUT /api/{resource} HTTP/1.1
Host: https://{api_base_url}/submissions
Authorization: Bearer {bearerToken} Ocp-Apim-
Subscription-Key {subscriptionKey}
```

Parameter	Description
bearer_token	The access token that is used to grant access to user-restricted APIs. This is provided at Step 3 in exchange for the authorisation code and will expire after 10 minutes. If necessary, you can generate a new one using the refresh token provided. Bearer tokens are issued as access tokens of 'bearer' type.
subscription_key	The subscription key provided to you by DfE. Please note that the subscription key will change depending on the DfE environment (supplier test or prod) you are accessing

Please refer to the API specification in the next section for the content to be provided as part of the submission process via API.

API specification for financial return submission

The API enables academy trusts to send their trial balance financial data to the Department using 3rd party software. The API allows a single financial upload to be made which is placed in a holding area within the online form. This will not automatically update information within the online form. For this to happen an authorised user must use the online form to consciously apply it to the form. When this happens, the data supplied by the API will be used to pre-populate the appropriate calls. In cases where the level of detail required by the form is less than that provided through the API, then the pre-population process will sum the lower level values and only enter the summary total.

If users wish to upload a revised data set, this is also supported and will result in the original values being over-written.

About testing the API

When software suppliers are enhancing their software product to use our APIs, we recognise it is important to ensure the testing process can simulate live operation. We support API testing in the following way:

- A separate end point and authorisation service is provided to minimise the risk of using the wrong service (i.e. removing the risk of using the live API for testing or the test API in live).
- Software suppliers are assigned a small list of dummy trusts which represent the different types of trust they support (i.e. MATs or SATs)
- User accounts will be created in DfE IDAMS for each dummy trust so that the software supplier can simulate the consent approval process as part of the testing.
- In addition to the API operation for submitting data there are two other REST methods which can be used to facilitate testing.
    - GET will allow the tester to re-read the file submitted and ensure it reflects the expected results
    - DELETE using submission GUID parameter will allow the tester to delete a submission before running the next test.
- For GET and DELETE endpoints, you will need to follow the same authorisation user journey to get the access token which further needs to be supplied as the bearer token as part of invoking the endpoints.

Draft trust financial statements

The draft trust financial statements functionality has been introduced so that trusts are able to use the same FMS data sets and API functionality to populate the trial balance elements of their statutory financial statements.

No changes are needed by trusts and software suppliers in the preparation and processing of the data into the online form, specifically:

- There is no change to the data requirements that trusts are required to provide for submission from FMS to the API

- There are no changes needed to the API, therefore, there no changes required to be made to the source data requirements or the process to import data into the online AR form via the API

- The draft trust financial statements file uses the same CoA version as the AR

- Prior year data columns within the workbook are populated from the final FMS data submission accepted into the AR form by the trust in the previous year, if they did not submit and accept data in the previous year, these columns will be blank. Therefore, FMS suppliers do not need to take any action to enable trusts to populate this data.

The draft trust financial statements will be available from the beginning of September for those trusts who can automate their data. Please review the timelines in the ‘Overview’ section of this document.

Data requirements for submission

1. Each academy within the trust is identified by its *Local Authority Establishment Number* and this needs to be provided as an "id" for Academy level data in the body of the data upload. For the avoidance of doubt each LAE number should only appear once in the list; also please note point 6 below.
2. Counterparty data items must be tagged and identifiable as *CounterpartyData*. If you are unable to send them, the counterpartyData element in the JSON can be set to either empty brackets {} or omitted entirely.
3. MAT central services data items must be tagged and identifiable as *matOverview*. If you do not have this data, the matOverview element in the JSON can be set to either empty brackets {} or omitted entirely.
4. Data must be provided to a maximum of 3 decimal places. Data values with fewer decimal places are acceptable but data values with more will cause an error to be returned.
5. Data must be provided at the academy trust level as a consolidated figure against a Trust UPIN or Company House Number.
6. Only provide data for the CoA items that have been populated in the FMS software. Where CoA items have not been populated in the FMS Software, omit these items from the API submission. Do not provide null, nil, blank or zero values. This ensures the AR form correctly informs the user of which items have been automated by the FMS API upload process. To note, zero is not an acceptable value where the academy has expressly entered this.
7. Only DfE account codes should be included in the data upload, therefore where the trust has added their own local account codes, these values must be consolidated into the DfE account code for that range, for example, any account codes added in the 855601 to 855609 range should be consolidated up to 855600 'Non Educational Contracts' for the FMS data upload. Also see 'CoA structure' heading below.
8. Where recent changes have been made in the composition of a Trust, the list of academies provided in the API submission should only include those for which financial transactions were recorded at any point during the past academic year ending on the 31st August. Hence, if an academy joined the trust on the 30th August, then it should be included in the submission, but if it joined on the 1st September then it should be excluded.
9. Source system must be provided while submitting the data. Example of source system – "XXX Financial Systems". This field is mandatory and is used for DfE internal reporting purposes (Registered software product name can be used)
10. The CoA version was "3.0.0" for AR 2021 collection. This can change every year depending on the changes made to the CoA codes. The CoA version will be updated accordingly.
11. If a trust wishes to take advantage of completing academy level tables, they must also submit data at central services and individual academy level.

12. If a trust has a trading subsidiary which is operating as part of the same legal corporate entity as the trust, this data will also need to be included in the API. For academy level data, the trust will need to decide whether to assign this to ’matOverview’ if the trading subsidiary applies to two or more academies, or ‘academies’ if the trading subsidiary is affiliated to one academy only.

There are 2 types of validation being applied to the request.

1. Json schema validation
2. Data validation

Refer to Appendix section – FMS API validations for the details of all the validations.

The data requirements for the AR and the draft trust financial statements are the same.

Data validations

The following validations are performed during the API submission process:

- check if the submission is valid Json, and that it does not violate the Json schema
- check if the coaVersion is valid, example `"5.0.0"`
- check if the “coaElement” in the coaElement:Value pair is valid for the coaVersion
- check if the “Value” in the coaElement:Value pair is valid and numeric value
- check if the trial balance total is zero, for Trust level data
- check that each academy referenced in the body of the upload is recognised by the DfE as belonging to the trust
- check if 0 or null is passed where the field is blank. No 0’s will be accepted
- check if any duplicate ids/keys are sent in the json, duplicate id/keys will not be accepted
- check if source system is not null. Value provided must be between 2 to 50 characters. It can contain letters, numbers, ‘&’ and ‘- ‘. No other characters will be accepted. This information will be used for internal DfE reporting purposes and enable us to support the automation process. We therefore ask that you ensure the value entered is recognisable as your FMS

FMS suppliers should capture below information in their application:

- Environment which they are connected to (supplier test, UAT, Production).
- API URL link and Token end point called from supplier application.
- Date/time when API calls are made.
- User details, if possible. (User ID, UPIN, Trust name)

Above details help in providing an audit trail for the FMS suppliers technical team to diagnose user errors.

FMS suppliers should refer users to API error codes available in this guide.

Data set information

There are four different types of data which can be sent via the API:

1. **matOverview –**
   This data is collected to pre-populate the MAT Central Services area of the AR benchmarking tab. This is an unconsolidated viewpoint and shows how the MAT is operating as a central standalone function, including transactions with academies that belong to the trust itself.
   MATs may provide services from their central trust to their academies – for example finance staff will be employed by the Trust but their work covers all academies. Often, but not always, this is funded by each academy in the trust

making a contribution, and it's up to the trust how they decide the allocation. Details of all the central income and expenditure, retained reserves and any centrally held assets need to be captured in the MAT Central Services tables within the benchmarking and land and buildings sections of the AR.
How you capture this is dependent on how your users normally account for central services. If they set up a separate entity for group/central services functions then it is this data we would ask for.

2. **Academies –**
   This is academy level trial balance data which is used to pre-populate the individual academy sections in the AR benchmarking and land and buildings sections. As with the "matOverview", this is an unconsolidated viewpoint and may include transactions with that MAT central services. A full trial balance for each academy can be sent and the AR form will pick out the data it requires.
3. **trustData –**
   This is the most important part of the submission which is used to pre-populate the majority of the AR form. This will be the full list of trial balance data at consolidated trust view and won't include any of the transactions within the trust itself, unlike the areas mentioned above. The trial balance data must total to zero or the submission will be rejected by the API.
4. **Counterpartydata –**
   This details any transactions with other Trusts in the sector totalled by account code and is used to prepopulate the counterparty section of the AR form.

Depending on how your users can set up their entities ("group" services in particular), it may be that trustData = academies + matOverview. Your product owners will be able to advise on this.

All four data sets are relevant for MATs to send. SATs do not need to send matOverview data as it's not applicable.

API version

There is only one version of the API live at any time. We are currently using API version 2.0.

API endpoints

Create or update Submission

Create or update Financial Return Submission via API.

The data submitted via this API can be used to pre-populate the online web form as part of the return preparation process.

The online form only holds a single API submission per collection period, thus subsequent calls to this API will overwrite the data previously uploaded for the current collection period.

Note that PUT should be called without a Submission ID included within the request data. The system will automatically create a Submission ID for a new submission or will automatically update the single existing submission for the period (there is only one submission per period - per trust).

PUT {api_base_url}/submissions

Parameters

Bearer token ***(Authorization header)***	Contains the access token received as part of the authorisation process.
Ocp-Apim-Subscription-Key * ***(header)***	Subscription key issued to the development organisation.
Enable-Dev-Error-Details ***(header)***	When set to true, error responses will provide additional details – note this parameter has no effect in the production environment.
Content-Type * ***(query)***	Identify the type of content being provided. Only “application/json” is supported.

Request body

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "definitions": {
    "coaItems": {
      "type": ["object"],
      "propertyNames": { "pattern": "^[0-9]{6}$" },
        "patternProperties": { "^[0-9]{6}$": { "type": "number", "multipleOf":
.001 }},
        "additionalProperties": false
    }
 },
 "type": "object",
 "properties": {
   "coaVersion": {"type": "string"},
   "trustData": { "$ref": "#/definitions/coaItems" },
   "counterpartyData": { "$ref": "#/definitions/coaItems" },
   "academyData": {
     "type": "object",
     "properties": {
       "matOverview": { "$ref": "#/definitions/coaItems" },
       "academies": {
       "type": ["object"],
       "propertyNames": { "pattern": "^[0-9]{3}-[0-9]{4}$" },
       "patternProperties": {"^[0-9]{3}-[0-9]{4}$":{"$ref":
       "#/definitions/coaItems"}},
       "additionalProperties": false },
      },
      "additionalProperties": false
    },
    "submittedBy": {"type": "string", "minLength": 2 },
    “sourceSystem”: {"type": "string", "minLength": 2,
                          "maxLength": 50 },
    "submissionType": {
      "type": "string",
      "enum": ["aar"]
    },
  },
  "additionalProperties": false,
  "required": [
    "coaVersion",
    "academyData",
    "trustData",
    "submittedBy",
    “sourceSystem”,
    "submissionType"
  ]
}
```

Schema

Field	Description
coaVersion	Chart of accounts version. This needs to be a valid CoA version and the CoA codes supplied within the data file must be included in that version.
submittedBy	Email address or user name of the FMS user providing the data.
sourceSystem	Source system from which the data is submitted. Value provided must be between 2-50 characters. It can contain letters, numbers, '&' and '- '. No other characters will be accepted. This information will be used for internal DfE reporting purposes and enable us to support the automation process. We therefore ask that you ensure the value entered is recognisable as your FMS.
trustData	Trust-level data. This will be a list of CoA codes plus their values as defined within the finance system.
counterpartyData	Counterparty data. This will be a list of CoA codes plus their values as defined within the finance system.
academyData	Covers academy data and is split into two sections: 1. MAT overview, 2. Academy-level data
submissionType	Identifies the type of data being submitted. Currently only the following are accepted; aar = Accounts return

Example body

The following test data can be used to test the API. Note the following:

1. This example includes data for two academies. The academies must belong to the trust from which consent is provided (covered by the company number /Trust UPIN parameter and the fact that the bearer token is from that trust). The data file needs to be updated with valid LA establishment codes for this academy in the format nnn-nnnn.
2. With effect from the AR1920 collection, the department is no longer gathering full counterparty data. Instead, the counterparty section is now focused solely on transactions within the academy sector. These transactions and balances (income, expenditure, debtors, creditors, other) with other academy trusts are to be reported in the counterpartyData object.

```
{
    "coaVersion": "3.0.0",
    "submittedBy": "fms.user@trust.co.uk",
    “sourceSystem”: “XXX financial system”
    "submissionType": "aar",
    "academyData": {
        "matOverview": {
            "125100": 20182.01,
            "125700": -20182.01,
            "880500": 0
        },
        "academies": {
            "925-3510": {
                "125100": 5098880,
                "880100": 2911.68,
                "880500": -5102106.18,
                "880550": 314.5
            },
            "925-2016": {
                "125100": 1919860.1,
                "880550": 746.91,
                "890200": 1152.57,
                "892100": -1921759.58
            }
        }
    },
    "trustData": {
        "125100": 10161663.87,
        "125700": -755883,
        "880500": -9444183.87,
        "892100": 38403
    },
    "counterpartyData": {
        "310100": 0,
        "310350": 74.62,
        "675450": 3906,
        "675460": 572.45,
        "675650": 5537.52,
        "675690": 344,
        "675700": 817,
        "860100": 5617
    }
}
```

The following embedded files provide further examples. Before using these files, the text ESTAB-1, ESTAB-2 and ESTAB-3 needs to be updated to identify academies which belong to the trust being used.

1. Test_Dataset_SAT.json represents a single academy trust and only contains one academy.
2. Test_Dataset_MAT.json represents a multi-academy trust and contains three academies.
3. PUT submission MAT minimal.json contains multiple academies but with minimal data.
4. PUT submission SAT minimal.json contains a single academy with minimal data.

Test_Dataset_SAT.json

Test_Dataset_MAT.json

PUT Submission SAT minimal.json

PUT Submission MAT minimal.json

Response codes

- 200 – Submission created successfully
  (Response body is a copy of the request, with the addition of submissionGuid & submittedDate (utc))
- 401/403 – Authentication failure
- 400 – Bad input request. The error response body is as follows:

```
{
    "type": (See Error Types in Data Validations section below),
    "details": (See details in Data Validations section below)
}
```

Refer to the Data Validations for scenarios of bad input request

Delete Submission (available for testing purposes only)

Delete an existing Financial Return submission against a Submission ID. The submission once deleted cannot be retrieved back.

DELETE {api_base_url}/submissions/{submissionGuid}

Parameters

Bearer token ***(Authorization header)***	Contains the access token received as part of the authorisation process.
Ocp-Apim-Subscription-Key * ***(header)***	Subscription key issued to the development organisation.
submissionGuid * ***(path)***	This is the submission ID of the last successful data upload. This will have been returned by a successful call to a PUT or GET operation.

Response codes

- 204 – Submission deleted successfully
- 400 – Bad input request. The response body shows more details as shown in the following example:

```
{
    "type": (See Error Types in Data Validations section below),
    "details": (See details in Data Validations section below)
}
```

- 403 – Forbidden request *(Examples include API request to delete a Submission which doesn't belong to the trust)*
- 404 – Submission not found *(Submission doesn't exist for the Submission ID)*

Get submission (available for testing purposes only)

Gets financial return submissions per trust.

GET {api_base_url}/submissions

Parameters

Bearer token ***(Authorization header)***	Contains the access token received as part of the authorisation process
Ocp-Apim-Subscription-Key * ***(header)***	Subscription key issued to the development organisation.

Response codes

- 200 – OK indicates the GET was successful. The following shows an example of the data returned:

```
{
      "coaVersion": "2.0.0",
      "submissionId": "5b6ca138-ff74-4f7c-baf1-9993b7ab4165",
      "submissionType": "aar",
      "submittedBy": "abc@academy.edu",
      "academyData": {
            "matOverview": {
                  "420100": 1100,
                  "460100": 1110.333,
                  "510100": 2200,
                  "510110": 1333,
                  "335250": -5743.333
            },
            "academies": {
                  "394-2091": {
                        "420100": 1000,
                        "460100": 2000,
                        "510100": 3000,
                        "510110": 1000,
                        "335250": -7000
                  },
                  "394-2112": {
                        "420100": 1000,
                        "460100": 2000,
                        "510100": 4000,
                        "510110": 1000,
                        "335250": -8000
                  }
            }
      },
          "counterpartyData": {
            "240100": 5600,
            "240200": 6200.33,
            "115706": 1100.99,
            "211100": 3300,
            "183100": 5800
      },
      "trustData": {
            "240100": 3000,
            "240200": 2000,
            "330100": -5500,
            "211100": 500
      }
}
```

- 204 – Submission not found
- 400 – Bad input request. The response body provides more details as the following example:

```
{
    "type": (See Error Types in Data Validations section below),
    "details": (See details in Data Validations section below)
}
```

Key environment variables

This section identifies the key parameters to access Financial Return Submission API for each independent DfE API environment. Software products which need to connect to multiple environments will need to be able to use the relevant parameters for that particular environment.

Note – You can get the subscription key for each environment using developer hub. The request needs an approval process in the developer hub and it can take up to 5 working days to get the key.

Supplier test

This environment is used by software suppliers to test their use of the API. Live data must NOT be used within this environment.

Parameter	Value
API base URL {api_base_url}	https://oat-api-customerengagement.platform.education.gov.uk/fmssuptest/api/
OAuth2.0 Authorize endpoint {authorize_base_url}	https://dfeb2cprod.b2clogin.com/dfeb2cprod.onmicrosoft.com/oauth2/v2.0/authorize?p=B2C_1A_consent_dfe_ste
OAuth2.0 Token endpoint {token_base_url}	https://dfeb2cprod.b2clogin.com/dfeb2cprod.onmicrosoft.com/oauth2/v2.0/token?p=B2C_1A_consent_dfe_ste
FMS API Scope URL {fms_api_scope_url}	https://dfeb2cprod.onmicrosoft.com/bfrp-ar/api.submit
Redirect URL	Supplied by the software supplier. Note that multiple URLs may need to be registered with the DfE if the software product needs to be tested from multiple environments.
Client ID	Supplied by the DfE. This is unique to a particular software product.
Client Secret	Supplied by the DfE. This is unique to a particular software product.
Subscription key	Supplied by the DfE. Note that separate subscription keys will be provided for different software products.
AR Form	https://s102t02-wa-01.azurewebsites.net/ar2122/

This environment will be available all year around.

User acceptance testing (UAT)

This environment is used by software suppliers during the user acceptance testing phase of the API. Live data must NOT be used within this environment. **This environment will be available from June-February.**

Parameter	Value
API base URL {api_base_url}	https://oat-api-customerengagement.platform.education.gov.uk/fmsuat/api/
OAuth2.0 Authorize endpoint {authorize_base_url}	https://dfeb2cprod.b2clogin.com/dfeb2cprod.onmicrosoft.com/oauth2/v2.0/authorize?p=B2C_1A_consent_dfe_uat
OAuth2.0 Token endpoint {token_base_url}	https://dfeb2cprod.b2clogin.com/dfeb2cprod.onmicrosoft.com/oauth2/v2.0/token?p=B2C_1A_consent_dfe_uat
FMS API Scope URL {fms_api_scope_url}	https://dfeb2cprod.onmicrosoft.com/bfrp-ar/api.submit
Redirect URL	Supplied by the software supplier. Note that multiple URLs may need to be registered with the DfE if the software product needs to be tested from multiple environments.
Client ID	Supplied by the DfE. This is unique to a particular software product.
Client Secret	Supplied by the DfE. This is unique to a particular software product.
Subscription key	Supplied by the DfE. Note that separate subscription keys will be provided for different software products.

The key environment variables for Production environment will be provided by DfE as the part of the cut-over to Live.

Chart of accounts (CoA)

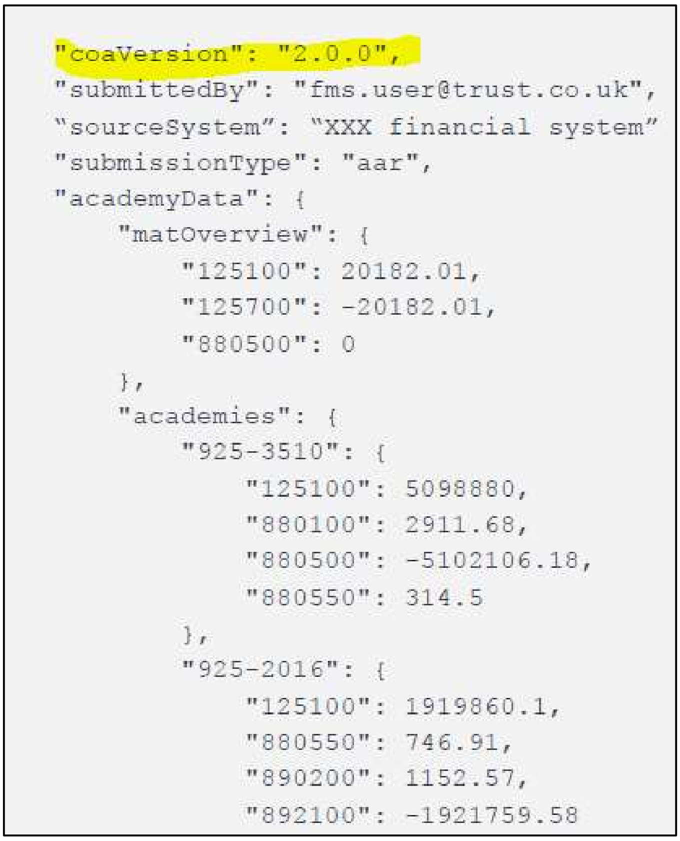
CoA version

The CoA version defines the Chart of Accounts (CoA) codes being used for the financial year. Version format example: 2.0.0

The same CoA version will also be used for the draft trust financial statements.

API Validations on CoA Version

Submissions via the API must include a “CoA Version”. See extract of a submission below:



```
"coaVersion": "2.0.0",
"submittedBy": "fms.user@trust.co.uk",
“sourceSystem”: “XXX financial system”
"submissionType": "aar",
"academyData": {
    "matOverview": {
        "125100": 20182.01,
        "125700": -20182.01,
        "880500": 0
    },
    "academies": {
        "925-3510": {
            "125100": 5098880,
            "880100": 2911.68,
            "880500": -5102106.18,
            "880550": 314.5
        },
        "925-2016": {
            "125100": 1919860.1,
            "880550": 746.91,
            "890200": 1152.57,
            "892100": -1921759.58
```

Format requirements

The API will reject any submission if the CoA Version does not meet the necessary format:

- It requires 3 numerical parts, separated by 2 dots. Each part has a range:
    - 1st value: a number between 1 and 99999. It cannot begin with a zero. For example, 1, 11, 111, 1111, 11111 are all ok. 0, 01, 000 are not ok.
    - 2nd value: a number between 0 and 99999. Examples of ok values: 0, 00, 001, 1111, 11111
    - 3rd value: a number between 0 and 99999. Examples of ok values: 0, 00, 001, 1111, 11111
- Acceptable formats: 1.0.0, 1.00.000, 12.01.1, 134.001.001
- Rejected formats: 01, 01.aaa.bbb, 1.9999999.222222222

Version validation

There is a metadata table in our database which lists all CoA uploads made. The most recent CoA is marked as 'active' and the rest as 'inactive'.

When a submission is made via the API, it will look up which CoA is marked as 'active' and compare this against what has been submitted. The API will check the 1st digit only. If this digit matches the 'active' CoA in the database, the submission will be permitted. Only the first digit is checked because variations of the second and third digits do not impact what account codes are sent via the API I.e., mapping changes (2nd digit change) or account description changes (3rd digit changes) will not impact the codes being sent.

CoA version change logic

CoA is published every year during spring on gov.uk. The CoA version can change depending on the changes made to the CoA for the financial year. We would avoid major changes whenever possible. The DfE team will communicate major changes, if any.

Revision	**CoA Change**
Major (1st digit)	- CoA code change
Minor (2nd digit)	- Mapping change - Signage adjustment change - Rounding account change - Subtotalling of accounts change
Maintenance – internal updates only (3rd digit)	- Description changes or reformatting

CoA account code structure

Account codes are made up of 6 digits, with the first 5 digits being pre-set by the DfE, and therefore the account codes laid out in the published CoA all end with a 0 (zero).

Using local account codes

The 6th digit may be used by a trust to set up 9 additional account codes against the DfE account code, therefore using 1 to 9 at the end of each DfE account code.

Trusts should not be able to alter/use the 5th digit of the account code, even if this has not been utilised by the DfE as:

- The automated mapping process will not pick these account codes up and the API will not be successful
- These are reserved for the DfE to add codes at a later date if needed

DfE ‘temporary’ account codes

The DfE have added 5 new ‘Temporary DfE’ account codes to the CoA 2023/24, to allow for any unforeseen grants that must be reported on by academy trusts in the accounts return.

These accounts codes are:

- 510950 - Temporary DfE / ESFA revenue grants 1 (for DfE use)
- 510980 - Temporary DfE / ESFA revenue grants 2 (for DfE use)
- 510990 – Temporary DfE / ESFA revenue grants 3 (for DfE use)
- 520400 - Temporary other Government revenue grants - not DfE (for DfE use)
- 550600 - Temporary DfE / ESFA capital grants (for DfE use)

Trusts should not have access to these account codes unless the DfE stipulates a use for them, and at that time, the description and mapping will be updated.

The DfE will inform trusts and FMS suppliers in the event that one of the above account codes is bought into use.

High level summary of CoA code ranges

CoA account code range	Description of account code range
100000 to 499999	Trial balance – Balance Sheet
500000 to 899999	Trial balance - Income and Expenditure (I&E)
900000 to 999999	Not trial balance - These are account codes allocated to additional information fields within the AR that are not linked to the trial balance, and therefore the mapping columns indicate the additional information trusts are required to provide over and above data that is pulled into the accounts return via automation ie FMS suppliers DO NOT need to build these into their trusts CoA

Glossary

Term	Description
Access token	Identifies a piece of information transmitted by a software system which contains the security credentials needed to call an API. The API can use the access token to check that the request is authorised.
API	Application Programming Interface: generic term for the method two computer systems use to interact.
AR	Accounts Return. This identifies a particular type of financial data collected by DfE once a year. This includes information from the trust financial accounts plus key supplementary information.
Bearer token	Generic term for the information transmitted during use of the API which identifies who is using the API. An access token is a specific example of a bearer token.
Chart of accounts	This identifies the set of financial codes which data needs to be supplied against. Financial software products may either hold data against these codes or may provide a mapping from an existing coding structure.
Client ID	This identifies a unique code assigned to the software product and is used with most OAUTH interactions to identify the software product.
Client secret	This identifies a “password” which is assigned to a software product which wishes to use the API. For certain interactions this can be used to confirm the identity of the software product being used. This should be updated annually.
OAUTH	Industry standard for securing APIs. The standard allows a user to approve the use of an API for particular purposes.
Redirect URL	Identifies an address or addresses supported by the third-party software system. This must be pre-registered and is used to return secure information to the software system.
REST	Representational State Transfer: this identifies a type of API which allows the state of data resources to be read and changed via create, update or delete methods.

Subscription key	This is a unique cost assigned to the developer or organisation. This must be passed during the API and is used as a secondary check of authorisation as well as for operational monitoring and support.

HTTP status codes

This section describes the typical response codes received from requests. In the case of errors this also indicates potential reasons for the error.

Code	Response body	Typical reasons
200	Submission data with the addition of the submission guid and datetime (utc) generated for this data.	Request was successful and the response typically shows the result.
204	N/A	Request is successful but no response is returned (for example successful deletion of a resource)
400	Application Error response	See Error Types Below
401	Access denied due to missing subscription key. Make sure to include subscription key when making requests to an API.	Ensure you have provided a subscription key which is valid for the API you are calling. This needs to be as a header called Ocp-Apim-Subscription-Key.
401	N/A	A valid bearer token was not provided.
403	N/A	While a valid bearer token was not provided it was not issued by a member of the trust for which data is being submitted.
404	Resource not found	The request was for a particular resource however that resource was not found. If this is for a particular submission ID, it is possible that the submission had previously been deleted. Alternatively, is the API path correct?
503	Service unavailable	The API is temporarily unavailable because it is overloaded or down for maintenance. Please wait for a short time and try again.

API error codes

Http error code 400 / Bad Requests will provide an application error response formatted as follows:

```
{
    "type": (error type),   <-- String identifying error types
    "details": (details)    <-- String array providing additional
details where applicable
}
```

The following table documents the possible error types:

Type	Details	Notes
coaValuePrecisionExceeds3dp	List of invalid CoA items	One or more CoA items has been provided with greater than three decimal places
invalidCoaCode	List of invalid CoA items	One or more unrecognised CoA items has been provided, or the CoA item is located in an invalid place
invalidFormSubmissionStatus	None	The trust is not yet setup by the DfE for accepting FMS submissions
invalidRequestBody	None	Invalid JSON data provided in request body
newFmsSubmissionsClosed	None	The trust has submitted their AR form data to the approval process, so FMS submissions are no longer accepted
requestBodyMissing	None	The request body is missing
schemaValidation	Dev only (see notes column)	Any json schema violation will trigger this error. During development, use of the Enable-Dev-Error-Details=true header will enable schema violation details in the response.
trustDataNotInBalance	None	Trust data provided within the request must balance.
unknownAcademy	List of unknown academies	During development, use of the Enable-Dev-Error-Details=true header will enable list of known academies in the response.
unknownCoaVersion	None	The provided CoA version is not known
unknownException	Dev Only	Any exception not covered above. During development, use of the Enable-Dev-Error-Details=true header will display technical error details.

AR202/23 release changes

There are no changes made to the consent process for the AR2022/23 release

List of changes in the submission API –

1. CoA Version for AR2022/23 is “5.0.0”

   This version number is subject to change prior to the release of the AR 2022/23. .There will be no new or deleted account codes in later versions, and changes would only apply to mapping, signage or descriptions.

Please refer to FMS API validations in Appendix section for details of all validations.

Refer to the document control section to review which areas of this guide have been impacted.

Document control

Version	Date Issued	Brief summary of change
0.1	24/05/2019	Initial draft
0.2	30/05/2019	Revised draft
0.3	07/06/2019	Update
0.4	11/7/2019	Revised with final code changes
0.5	Nov 2019	Updated
0.6	22/04/2020	Enhanced to clarify operation of API.
0.7	23/04/2020	• Clarified the operation of the API. • Improved description of parameters. • Added more detailed description of errors.
0.8	24/04/2020	Performed various cosmetic updates following review by TC.
0.9	27/04/2020	Performed updates following review by AB, JH & AS
1.0	01/05/2020	• Clarified selected explanation text • Added glossary. • Clarified position regarding counterparties for next release of API • Update API endpoint to new location
1.0.1	06/05/2020	Removed tracked changes and further clarified use of multiple URLs and number of decimal places.
1.0.2	07/05/2020	Correct couple of typos
1.1.0	26/06/2020	API version 1.1 request and error response updates
1.1.1	16/07/2020	Updated return codes and error types
1.1.2	23/07/2020	Updated to include OAuth 2.0 process; Update API to remove the need to provide Company Number as query parameter. Merged the two documents issued most recently on Thursday 16th July 2020.
1.1.3	07/08/2020	Updated the api_base_url in the key environment variable section Replaced authorisation_base_url with two separate authorise_base_url and token_base_url. Added environment key variables for UAT

Version	Date Issued	Brief summary of change
1.1.4	11/08/2020	Added unknownException API error type Step 1: Invoke authorisation – updated scope to include additional parameter {fms_api_scope_url} Step 3 & 4: Added additional code for scope parameter Updated environment key variables for {fms_api_scope_url}
1.1.5	18/08/2020	Updated value of the scope in step 3: Exchange authorisation code for access token
1.1.6	22/10/2020	Added numbering scheme for the section "Data Requirements for Submission" Added points 4,5 and 6 to the section "Data Requirements for Submission"
2.0.0	31/03/2021	A new version for the AR2021 collection – Sections updated – Registering your software product Data requirements for Submission Data Validation Request Body – Schema, Example body and Sample json AR2021 Release changes Appendix – FMS API Validations
2.1.0	14/09/2021	Sections updated – Registering your software product Key Environment Variables API Version New Sections added - Data Set Information CoA Version
3.0.0	11/03/2022	Sections updated - • Data requirements • Data set information • Release changes

Version	Date Issued	Brief summary of change
4.0.0	30/06/2022	Sections updated – • Overview – new sections for: ○ Timelines ○ Draft trust financial statements • Data requirements update – points added for: ○ MAT central services ○ DfE account codes ○ Trading subsidiaries *Note: these are not new requirements, but added for clarity* • Step 1: Invoke the Authorisation endpoint – updated consent statement screen shot • AR2022 release changes New Sections added - • Draft trust financial statements
5.0.0	*24/05/2023*	New section added – • Annual update of client secret ID's
5.0.0	*24/05/2023*	New section added – • CoA account code structure

Appendix

FMS API validations

The following explains the validation and responses of the json request data that is sent via the API.

Validation errors are returned as http status code 400 “Bad request”. The response body will be a json message. It has a field called “type” which is used to indicate what type of error it is returning.

Example: The following is a schema validation error where zero was passed as a value.

```
{
    "type": "schemaValidation"
}
```

In development, a http header can be set which may result in a more detailed explanation of the error.

http header: Enable-Dev-Error-Details=true/false

With the header set to true the json response now looks like this:

```
{
    "type": "schemaValidation",
    "details": [
        "Path 'academyData.academies.925-
2016.125100', Line: 9, Position 27, Error: Value 0 is not allowed"
    ]
}
```

There are 2 types of validation being applied to the request.

1. Json schema validation
2. Data validation

They are evaluated separately. This means if a json schema validation error is found, it will not be passed to the data validation. As a result, errors are group by type.

If the request has schema errors, then it is considered ill formed and as such data validation may give misleading results.

Json schema validation

In this validation, a schema is used to specify the general format of the data. What fields are allowed, the lengths, upper/lower limits, which fields are mandatory etc. We refer to this as structural/format validation.

The following are validations at this level.

- Duplicate key

  For the most part, json can be considered as a series of key/value pairs. The key represents a property, and the value is the value for the property.

  Example. `{"name": "Bugs Bunny", "age": 30}`

  The above json has 2 keys name and age.

  An example of duplicate keys: `{"name": "Bugs Bunny", "name": "Elma Fudd", "age": 30}`

  This has 2 “name” keys, which is used to specify the name? Why does it have 2 names? Is that intentional, a mistake? People will have different views. Given it is therefore not clear what the meaning is, duplicate keys are not allowed.

  With the dev error details enabled, the error will tell you which key and the location.

```
{
    "type": "schemaValidation",
    "details": [
        "Path 'academyData.academies.925-
2016', Line: 11, Position 23, Error: Property with the name '925-
2016' already exists in the current JSON object."
    ]
}
```

  Current limitations: Only the details of the first duplicate are returned. This is only an issue with dev errors enabled as errors messages are not included in when not enabled.

- Zero values

  All values must be supplied as either a negative or positive number. No values can be passed as zero or variations of.

  The following are all “zero”.

  - 0
  - 0.0
  - 0.00

  Example:

  Input: `"925-2016": {"125100": 0}`

Response (with dev errors enabled):

```
{
    "type": "schemaValidation",

    "details": ["Path 'academyData.academies.925-
2016.125100', Line: 9, Position 27, Error: Value 0 is not allowed"
]
}
```

- Upper/lower limits

  All numerical values need to be with in a specific numeric range. Any values outside of this range will trigger an error.

  Allowed range: -999999 and 999999 (excluding zero)

  Example:

  Input: `"925-2016": {"125100": -99999999}`

  Response (with dev errors enabled):

```
{
    "type": "schemaValidation",

    "details": [
        "Path 'academyData.academies.925-
2016.125100', Line: 9, Position 35, Error: Integer -
99999999 is less than minimum value of -999999. Integer -
99999999 is less than minimum value of 0."
    ]
}
```

  Input: `"925-2016": {"125100": 99999999}`

  Response (with dev errors enabled):

```
{
    "type": "schemaValidation",
    "details": [
        "Path 'academyData.academies.925-
2016.125100', Line: 9, Position 34, Error: Integer 99999999 exceed
s maximum value of 0. Integer 99999999 exceeds maximum value of 99
9999."
    ]
}
```

  Current limitations: The way ranges are implemented in the schema creates 2 different messages based on the value passed. Both refer to exceeding 2 numbers. This is being looked at to create a simpler error. This is only an issue with dev errors enabled as errors messages are not included in when not enabled.

- Number of decimal places

  All numeric values must be between 0 and 3 decimal places.

  Valid values example: `1, 2.0, 3.14, 6.666, 2.000`
  Invalid values example: `1.0000, 2.0000000, 3.141592, 6.6661`
  Input: `"925-2016": {"125100": 1.0001}`

  Response (with dev errors enabled):

```
{
    "type": "schemaValidation",

    "details": [
        "Path 'academyData.academies.925-
2016.125100', Line: 9, Position 32, Error: Float 1.0001 exceeds ma
ximum value of 0. Float 1.0001 is not a multiple of 0.001."
    ]
}
```

  Not a multiple of .001 simply means it exceed 3 decimal places.

  The only exception is when the number of decimals exceeds 3 but the value can be changed to 3 without altering the value. This means no loss of data through rounding occurs.

  Examples:
  `1.0000000` can be reduced to `1`
  `1.0010000` can be reduced to `1.001`

- Mandatory fields

  Certain fields have been marked as mandatory. Failure to send them will result in a schema validation error.

  Example, the `coaVersion` key is required. I the json request does not send this data, the following error is returned.

  Input (missing coaVersion): "`{"academyData": { … } }`

  Response (with dev errors enabled):

```
{
    "type": "schemaValidation",
    "details": ["Path '', Line: 1, Position 1, Error: Required pro
perties are missing from object: coaVersion."]
}
```

The current mandatory fields are:

- coaVersion
- academyData
- trustData
- submittedBy
- submissionType
- sourceSystem

- Regular expressions (Regex)

Several fields are checked to ensure they match certain lexical patterns. This is performed using a “regular expression”. The expressions check length / characters used etc.

The current expressions used are:

<table>
<tr><th>Field</th><th>Pattern</th><th>Meaning</th><th>Example</th></tr>
<tr><td>Academy LAE</td><td>^[0-9]{3}-[0-9]{4}$</td><td>3 digits followed by a dash followed by 4 digits. Digits allowed are 0 to 9. Nothing else can be part of the value.</td><td>Good: 123-4567
Bad: A23d-44er</td></tr>
<tr><td>COA Code</td><td>^[0-9]{6}$</td><td>6 digits. Digits allowed are 0 to 9. Nothing else can be part of the value.</td><td>Good: 123456
Bad: A23d-44er</td></tr>
<tr><td>Submitted by</td><td>^[^<>\\[\\]{}/'#:!=|&+*?^$]+$</td><td>None of the characters in the expression.</td><td>Need to verify this</td></tr>
<tr><td>Source system</td><td>^[a-zA-Z0-9-\\&\\.]+$</td><td>One or more characters. Characters allowed are A to Z (upper and lower case), Dash (-), Ampersand (&), Period (.) or a space.</td><td>Good: Acme & Sons FMS V1.2
Bad: Acme #V1.2 $$$</td></tr>
</table>

- Unknown / additional keys

  The request must not send any additional keys not already defined in the schema. The only exception is the academy LAE numbers and COA codes.

  This means you cannot send extra information as the processor is not expecting it.

  The following is the expected structure.

```
{
    "coaVersion": "2.0.0",
    "academyData": {
        "matOverview": {
            "115100": 50.00
        },
        "academies": {
            "925-2016": {
                "125100": 1.01
            }
        }
    },
    "trustData": {
        "125100": 1,
        "125200": -1
    },
    "submittedBy": "TestUser",
    "submissionType": "aar",
    "sourceSystem": "DfE FMS Supplier System"
}
```

  Example: Adding a coaDescription key.

  Input:

```
{
    "coaVersion": "2.0.0",
    "coaDescription": "Some description text",
    "academyData": {…},
    "trustData": {…},
    "submittedBy": "TestUser",
    "submissionType": "aar",
    "sourceSystem": "DfE FMS Supplier System"
}
```

  Response (with dev errors enabled):

```
{
    "type": "schemaValidation",
    "details": [
        "Path 'coaDescription', Line: 3, Position 21, Error: Property 'coaDescription' has not been defined and the schema does not allow additional properties."
    ]
}
```

All schema validation errors are returned with a type of “schemaValidation”.

Data validation

In this validation, the actual values are validated. This group of validations is only checked once the json schema validation is successful. Note, there are several checks here that are now covered by the json schema check and will therefore not be executed unless the data validation is executed manually.

- CoA Version

  The coa version is checked to see if it matches an entry in the database. Passing a version that is not supported will result in an unknown coa version error.

  Input:

  ```
  {
      "coaVersion": "9999.0.0",
      "academyData": {…},
      "trustData": {…},
      "submittedBy": "TestUser",
      "submissionType": "aar",
      "sourceSystem": "DfE FMS Supplier System"
  }
  ```

  Response:

  ```
  {
      "type": "unknownCoaVersion"
  }
  ```

- CoA codes

  The coa codes are checked to ensure they are supported. Passing a code that is not supported will result in an unknown coa code error.

  Input:

  ```
  {
      "coaVersion": "~2",
      "academyData": {
          "matOverview": { … },
          "academies": {
              "925-2016": {
                  "888888": 1.01
              }
          }
      },
      "trustData": { … },
      "submittedBy": "TestUser3",
      "submissionType": "aar",
      "sourceSystem": "DfE FMS Supplier System"
  }
  ```

Response (with dev errors enabled):

```
{
    "type": "invalidCoaCode",
    "details": [
        "925-2016.888888"
    ]
}
```

- CoA value data type

  The values passed with the coa codes are checked to ensure they are numeric. Passing a non-numeric value result in an invalid value type error.

  Note: This check is covered by the json schema validation.

  Input:

```
{
    "coaVersion": "~2",
    "academyData": {
        "matOverview": { … },
        "academies": {
            "925-2016": {
                "125100": "1.01"
            }
        }
    },
    "trustData": { … },
    "submittedBy": "TestUser3",
    "submissionType": "aar",
    "sourceSystem": "DfE FMS Supplier System"
}
```

  Response (with dev errors enabled):

```
{
    "type": "coaValueInvalidType",
    "details": [
        "925-2016.125100"
    ]
}
```

- CoA value decimal places check

  The values passed with the coa codes are checked to ensure they have no more than 3 decimal places. Passing a value with more than 3 decimal places will result in a precision exceeded error.

  Note: This check is covered by the json schema validation.

Input:

```
{
    "coaVersion": "~2",
    "academyData": {
        "matOverview": { … },
        "academies": {
            "925-2016": {
                "125100": 1.000001
            }
        }
    },
    "trustData": { … },
    "submittedBy": "TestUser3",
    "submissionType": "aar",
    "sourceSystem": "DfE FMS Supplier System"
}
```

Response (with dev errors enabled):

```
{
    "type": "coaValuePrecisionExceeds3dp",
    "details": [
        "925-2016.125100"
    ]
}
```

- Trust data balance check

  The values in the trust data section need to balance. In order to balance, the values must total zero. All values where their coa code does not start with a 9 are involved.

  In the following example, only the first 2 codes are used and the 3rd is excluded from the sum. The result is zero as 1 + -1 = 0

```
"trustData": {
    "125100": 1,
    "125200": -1,
    "900100": 1
}
```

Input:

```
{
    "coaVersion": "~2",
    "academyData": { … },
    "trustData": {
        "125100": 1,
        "125200": 1,
        "900100": 1
    },
    "submittedBy": "TestUser3",
    "submissionType": "aar",
    "sourceSystem": "DfE FMS Supplier System"
}
```

Response:

```
{
    "type": "trustDataNotInBalance"
}
```

In the above example, the 3rd value is excluded, and the first 2 fields are used. However, the total is 2 (1 + 1) and not zero.

API Details

UAT Environment:

- UAT API URL : https://oat-api-customerengagement.platform.education.gov.uk/fmsuat/api/Submissions
- UAT Token End point : https://dfeb2cprod.b2clogin.com/dfeb2cprod.onmicrosoft.com/oauth2/v2.0/token?p=b2c_1a_consent_dfe_uat

STE Environment:

- STE API URL : https://oat-api-customerengagement.platform.education.gov.uk/fmssuptest/api/submissions
- STE Token End point : https://dfeb2cprod.b2clogin.com/dfeb2cprod.onmicrosoft.com/oauth2/v2.0/token?p=B2C_1A_consent_dfe_ste

Prod Environment:

- Prod API URL : https://education-financial-reporting.api.gov.uk/fmsprod/api/submissions
- Prod Token End point: https://dfeb2cprod.b2clogin.com/customeridentity.education.gov.uk/oauth2/v2.0/token?p=B2C_1A_consent_dfe