Version 1.0, 30 September 2019

REF2021 Submission System
API Gateway

REF2021 Research Excellence Framework Submission System

User Guide to the API endpoints

Copyright and Contact

For further information, please contact the following:

REF 2021, Nicholson House, Lime Kiln Close, Stoke Gifford, Bristol, BS34 8SR

Telephone: 0117 905 7630

Email: **usersupport@ref.ac.uk**

CONTENTS

CHAPTER 1

Introduction

What is the API Gateway?

Underpinning the REF2021 submission system is a RESTful core service API, with a series of endpoints that correspond to the actions described in the data entry forms. These endpoints are collectively referred to as the *API Gateway*.

All of the endpoints can be accessed using **Swagger UI**. There is a separate set of endpoints on **Swagger UI** for each of the following two systems:

- The UAT system at: **https://testsubmissionsapi.ref.ac.uk/swagger/index.html**.
- The live system at: **https://submissionsapi.ref.ac.uk/swagger/index.html**.

Swagger UI allows you to expand the endpoints to view the associated parameters and examples, and to try them out by generating responses to specific requests. Examples of the use of **Swagger UI** are given in the sections following on from **Making API requests using Swagger UI**.

CHAPTER 2
Initial setup

Before you start

Before you can use any of the endpoints in the API Gateway, you need an **Automated user** account in the REF2021 submission system, with the requisite **User functions** (for example, **Import** and **Submission management**) and the appropriate **User permissions**. If you do not have such an account, ask the system administrator at your HEI to create one for you. Then make a note of the **API Key** associated with the account. This key will need to be supplied in the header of every API call that you make.

For example, if the **API Key** is `o6kQWgcajOVUg2BSfnJUwXBeCf4CMjUdcUKDRZUm`, and you want to request the list of import jobs, then the header for the `GET` request might start as follows:

```
GET /api/importjobs HTTP/1.1
ref2021-apikey: o6kQWgcajOVUg2BSfnJUwXBeCf4CMjUdcUKDRZUm
```

For the use of the **API Key** with **Swagger UI**, refer to **Authorisation**.

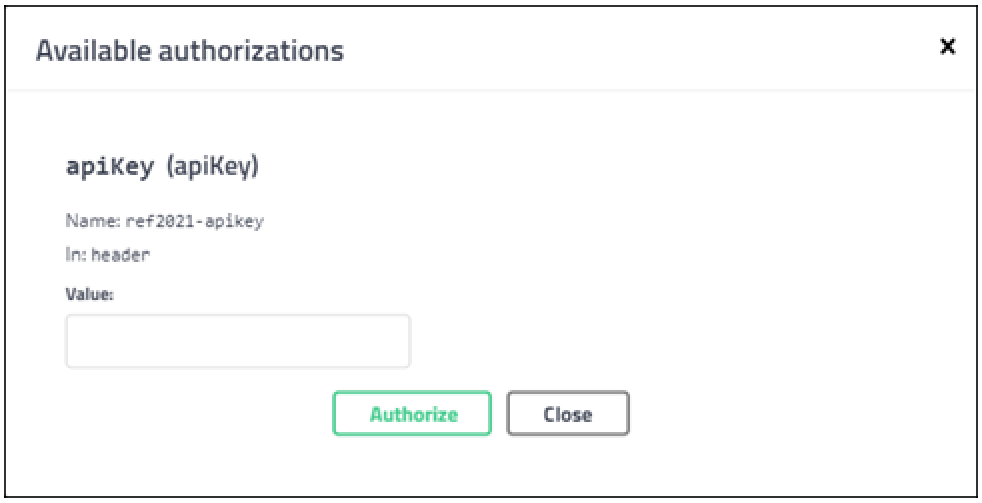
CHAPTER 3

Using Swagger UI

Making API requests using Swagger UI

Authorisation

At the beginning of the session in which you are going to make API requests, click the **Authorize** button on the top right of the **Swagger UI** screen, to display the following dialog:


Available authorizations

apiKey (apiKey)

Name: ref2021-apikey

In: header

Value:

In the **Value** box, enter the value of the **API Key** associated with the user account for which the API is to be run, and then click **Authorize**, followed by **Close**.

If you do not have a user account, refer to Before you start and be sure to make a note of the **API Key** associated with the account.

Generating responses to specific requests

Let us suppose that we wish to request the submissions that have been created, verify the numbers of doctoral degrees that have been entered for a particular submission, update the numbers for this submission using the API, and, finally, clear the numbers of Doctoral Degrees for the same submission.

If you are going to use any of the HTTP verbs other than *GET* (for example if you are going to use the API to *PUT*, *POST*, *PATCH* or *DELETE* data), remember that you will be modifying the current data in the associated submission system. Therefore, although we will be using the UAT system, it is still advisable to get into the habit of clearing any unwanted data, before you move from exploring and learning about the API in the UAT system, to using the API in the live system.

In most of the requests to the API, you do not need to include all the endpoint parameters listed in the **Swagger UI** documentation. For example, the *auditLog* properties are not required as the API will add them automatically when processing the *PUT*, *POST*, *PATCH* or *DELETE* request, or in the case of a *GET* request, returning the requested details.

1. Go to the UAT system: **https://testsubmissionsapi.ref.ac.uk/swagger/index.html**.
2. Under **Lookups**, click `GET /api/lookup/institutions` to expand the endpoint that returns the IDs and names of the HEIs.

3. Click **Try it out**.
4. Select the required **Response content type** from the drop-down, for example `application/json`.
5. Click **Execute** to submit the request and show the *cURL* that was submitted, for example.

```
curl -X GET "https://testsubmissionsapi.ref.ac.uk/api/lookup/institutions" -H
"accept: application/json" -H "ref2021-apikey:
o6kQWgcajOVUg2BSfnJUwXBeCf4CMjUdcUKDRZUm"
```

Note the use of the **API Key** that was entered at the start of the session, in the header.

6. If the request was successful, copy and paste the content that is returned in the **Response body**, so that you can use the institution codes in other requests.
7. Click **Clear** followed by **Cancel**.
8. Click on the endpoint to collapse it again.
9. Under **Submissions**, click `GET /api/submissions` to expand the endpoint that returns the created submissions.
10. Click **Try it out**.
11. Enter any parameters that are prompted for (in this case, the code of the HEI, from those returned at step 6).
12. Select the required **Response content type** from the drop-down.
13. Click **Execute** to submit the request and show the *cURL* that was submitted.
14. If the request was successful, copy and paste the content that is returned in the **Response body**, so that you can use the `submissionIDs` in other requests.
15. Click **Clear** followed by **Cancel**.
16. Click on the endpoint to collapse it again.
17. Under **DoctoralDegreesAwarded**, click `GET /api/submissions/{submissionId}/doctoralsawarded` to expand the endpoint that returns the numbers of doctoral degrees awarded for a particular submission.
18. Click **Try it out**.
19. Enter any parameters that are prompted for (in this case, the `submissionID`, from the IDs returned at step 14).
20. Select the required **Response content type**.
21. Click **Execute** to submit the request and show the *cURL* that was submitted.
22. Verify the content that is returned in the **Response body**, and make a note of the *userID* and update information that you wish to use to modify the entries using the *PUT* statement, for example:

```
"updatedByUserId": "8f702d26-8e9e-45ba-863d-6474f12098f1",
"updatedBy": "Chester Admin",
"lastUpdatedOn": "2019-06-28T11:54:11.974Z"
```

23. Click **Clear** followed by **Cancel**.
24. Click on the endpoint to collapse it again.
25. Under **DoctoralDegreesAwarded**, click `PUT /api/submissions/{submissionId}/doctoralsawarded` to expand the endpoint that allows you to update the numbers of doctoral degrees awarded for a particular submission.

26. Click **Try it out**.
27. Enter the `submissionID`, from the IDs returned at step 14.
28. Update the *Degrees Awarded* model as in the following example:

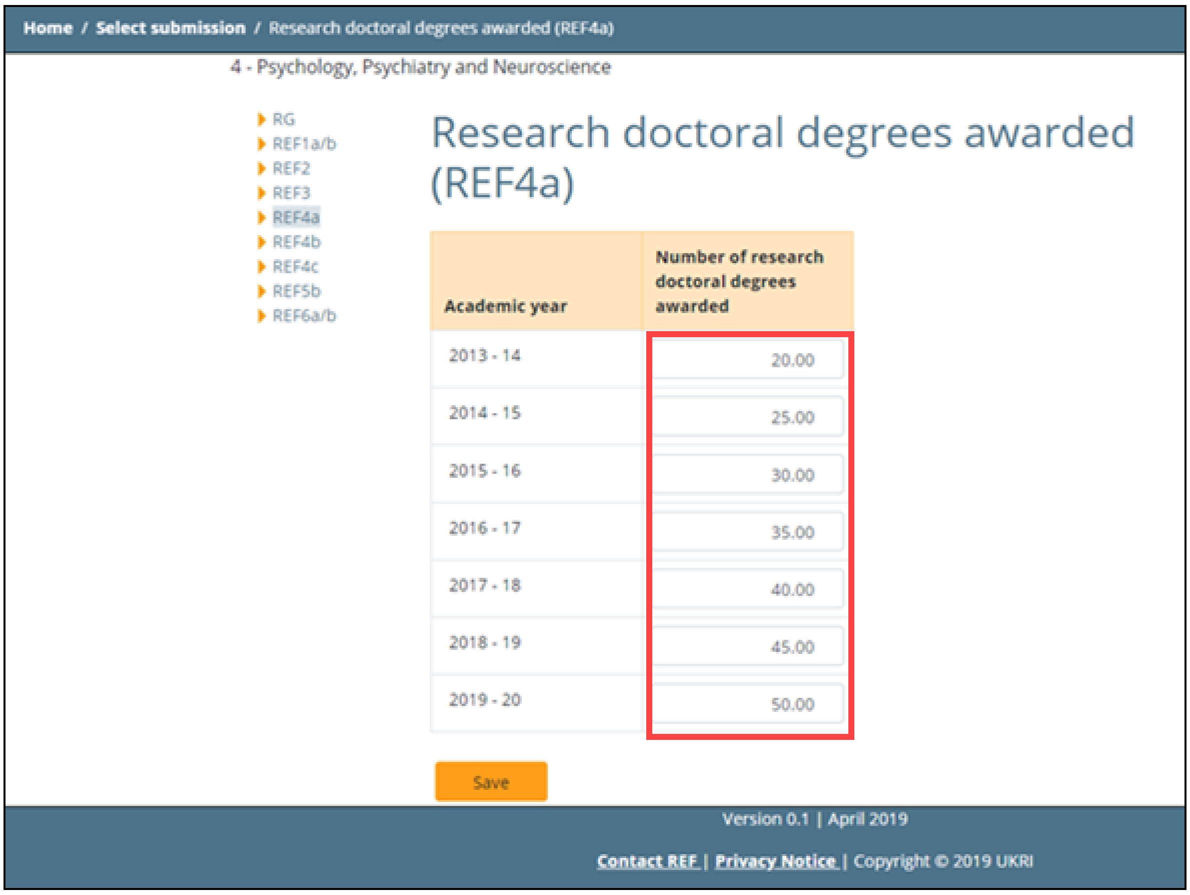
```
{
  "submissionId": "0004efa1-c5a3-4a60-b9fb-3278bcdfd0d8",
  "year2013": 20,
  "year2014": 25,
  "year2015": 30,
  "year2016": 35,
  "year2017": 40,
  "year2018": 45,
  "year2019": 50,
  "versionNumber": 0,
  "processingContext": {},
}
```

29. Select the required **Response content type**.
30. Click **Execute** to submit the request and show the *cURL* that was submitted.
31. Verify the content that is returned in the **Response body**, for example:

```
{
  "value": {
    "submissionId": "0004efa1-c5a3-4a60-b9fb-3278bcdfd0d8",
    "year2013": 20,
    "year2014": 25,
    "year2015": 30,
    "year2016": 35,
    "year2017": 40,
    "year2018": 45,
    "year2019": 50,
    "versionNumber": 1,
    "processingContext": null,
    "auditLog": null,
    "lastUpdatedByUserId": "9fccb63f-5c67-4a19-bb33-adcf6471b34f",
    "lastUpdatedBy": "JohnRobertGreen",
    "lastUpdatedOn": "2019-06-28T09:09:20.04Z"
  },
  "isSuccessful": true,
  "isFatal": false,
  "errors": []
}
```

Note that the user and update information has been modified automatically in accordance with the user executing the API.

32. You can now verify that the corresponding data entry screen has been updated:

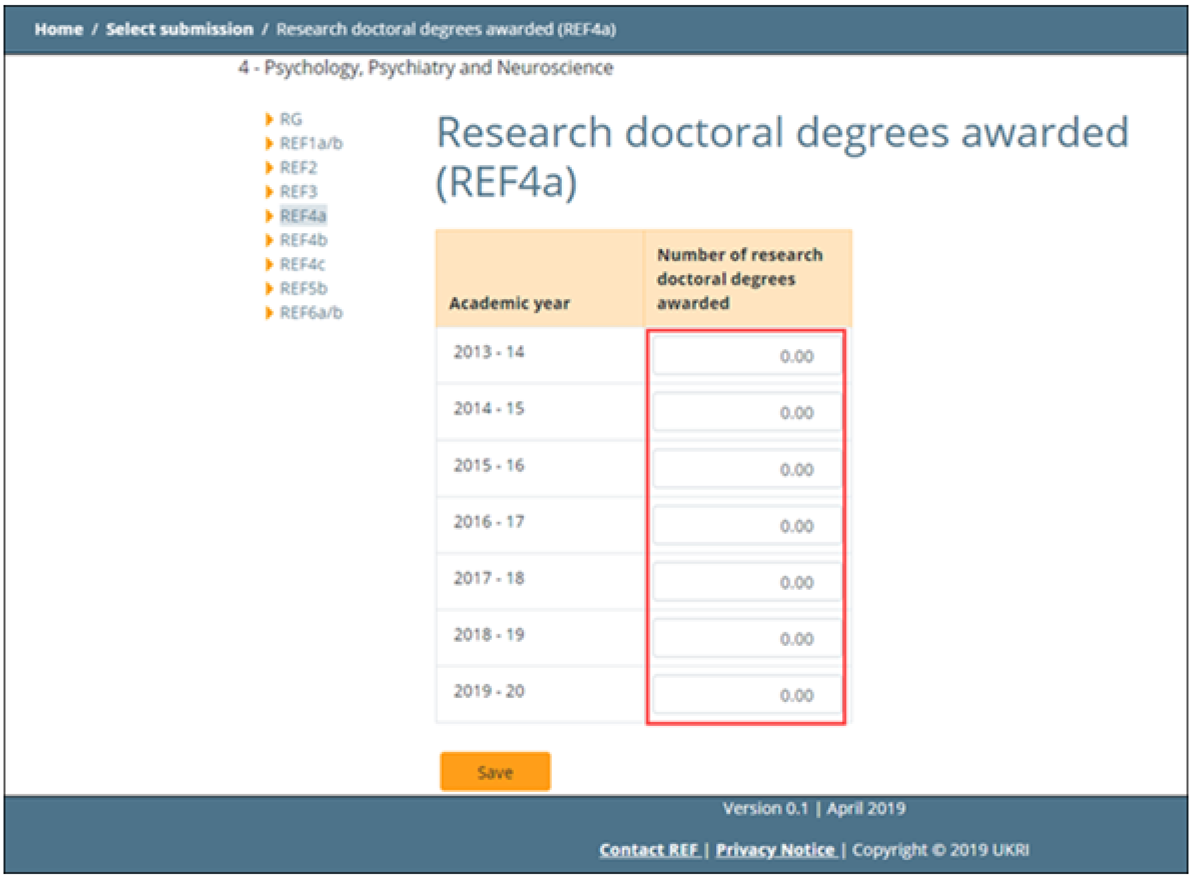

Home / Select submission / Research doctoral degrees awarded (REF4a)
RG
REF1a/b
REF2
REF3
REF4a
REF4b
REF4c
REF5b
REF6a/b
Save
Version 0.1 | April 2019
Contact REF | Privacy Notice | Copyright © 2019 UKRI

4 - Psychology, Psychiatry and Neuroscience

Research doctoral degrees awarded (REF4a)

Academic year	Number of research doctoral degrees awarded
2013 - 14	20.00
2014 - 15	25.00
2015 - 16	30.00
2016 - 17	35.00
2017 - 18	40.00
2018 - 19	45.00
2019 - 20	50.00

33. Click **Clear** followed by **Cancel**.
34. Click on the endpoint to collapse it again.
35. Under **DoctoralDegreesAwarded**, click `DELETE /api/submissions/{submissionId}/doctoralsawarded` to expand the endpoint that allows you to clear the numbers of doctoral degrees awarded for a particular submission.
36. Click **Try it out**.
37. Enter the `submissionID`, from the IDs returned at step 14.
38. Select the required **Response content type**.
39. Click **Execute** to submit the request and show the *cURL* that was submitted.
40. You can now verify that the corresponding data entry screen has been 'nulled':



Home / Select submission / Research doctoral degrees awarded (REF4a)

4 - Psychology, Psychiatry and Neuroscience

Research doctoral degrees awarded (REF4a)

Academic year	Number of research doctoral degrees awarded
2013 - 14	0.00
2014 - 15	0.00
2015 - 16	0.00
2016 - 17	0.00
2017 - 18	0.00
2018 - 19	0.00
2019 - 20	0.00

Version 0.1 | April 2019

Contact REF | Privacy Notice | Copyright © 2019 UKRI

41. Click **Clear** followed by **Cancel**.
42. Click on the endpoint to collapse it again.
43. Log out of the session by clicking the **Authorize** button on the top right, followed by **Logout** and then **Close**.

Making an API import request

You can replicate the GUI import process using the API endpoint `POST /api/importjobs`. The basic steps are the same for each file format, that is to say:

1. Ensure that you have an import file in the relevant format (`xlsx`, `XML` or `JSON`) that contains all the records to be imported.

   Guidance on the process, together with schemas and examples in the different formats, are available from the **Submission system data requirements** web page. You are advised to take a copy of the examples and adapt them for your own use.

   The API endpoint `POST /api/importjobs` can only currently be used to import Research Groups and REF6a/b data.

2. It is a requirement that the contents of the file are BASE64-encoded, for reasons of compression and consistency. Therefore, use an appropriate encoder (for example, **https://base64.guru/converter/encode/file** or **https://www.browserling.com/tools/file-to-base64**) to encode your import file, and paste the result somewhere handy, so that it can be entered into the `fileContent` field in the endpoint *Import Job* model.
3. Go to the UAT system: **https://testsubmissionsapi.ref.ac.uk/swagger/index.html**.
4. Enter the value of your **API Key** in accordance with **Authorisation**.
5. Under **ImportJobs**, click `POST /api/importjobs` to expand the endpoint that allows you to create an import job.
6. Click **Try it out**.

Remember that you will be modifying the current data in the associated submission system. Therefore, although we will be using the UAT system in the examples, it is still advisable to get into the habit of clearing any unwanted data, before you move to exploring and learning about the API in the live submission system.

7. Update the *Import Job* model in accordance with the required example from the following sub-topics:
   - **Importing records from an Excel file**
   - **Importing records from an XML file**
   - **Importing records from a JSON file**
   - **Replacing previously imported records**

   The model contains the following parameters, which correspond to the fields described in the GUI Import process.

API import parameters

API Parameter	Equivalent GUI Field	Possible API values (case-insensitive)
`fileFormat`	**File format**	Needs to match the format of the file being imported, that is to say: `ExcelXlsx`, `XML` or `JSON`.

API Parameter	Equivalent GUI Field	Possible API values (case-insensitive)
`fileName`	**File to import** and then **Choose file**.	Supply only the name of the import file, and not the path. The file suffix is optional as this is determined by the `fileFormat`. The content of the file will be extracted from the **fileContent**.
`reportFileFormat`	**Report format**	Either `JSON` or `ExcelXlsx`.
`importMode`	**Update preference**	Either `Merge` or `Replace`. ⚠ Take care when using the `Replace` option, as the presence of the smallest amount of data in the import file for a UOA will result in **all** the existing data for that UOA being overwritten.
`importWithErrors`	**Allow data to be imported even if there are validation errors**	Either `true` or `false`.
`retrieveCitationCounts`	**Retrieve citation counts for outputs**	Ensure that this is set to `false` for the time being.
`zippedFile`	**Zipped file**	Either `true` or `false`, depending on whether or not the import file is zipped.
`fileContent`	**N/A**	A BASE64-encoded string of the file contents. You will need to find a tool that can accept a binary file and create a BASE64 string from it (for example, **https://base64.guru/converter/encode/file** or **https://www.browserling.com/tools/file-to-base64**).
`tags`	**N/A**	The `tags` parameter is mandatory for importing, and, if you omit it, then `"tags": "import"` will be supplied by default. If you enter your own `tags`, then you should separate them by semi-colons, but at least one of them should be `"import"`. ℹ The `"import"` tag is necessary in order for the associated job to appear in the user's list of import jobs, and for the submission system to be updated.
`outputFilename`	**Report filename**	The filename for the validation report. ℹ Do not supply the file extension, as the import engine will add an appropriate file extension.

> In order to create an import job, you do not need to include all the endpoint parameters listed in the **Swagger UI** documentation, only those listed above. For example, the *auditLog* and *log* properties are not required as the API will add them automatically.

8. Select the required **Response content type** from the drop-down, for example `application/json`.
9. Click **Execute** to submit the request and show the *cURL* that was submitted.
10. Verify the content returned in the **Response body**, and ensure that `isSuccessful` is set to `true`.

> The user and update information is modified automatically in accordance with the user executing the API.

11. You can now verify the results in the GUI by looking at the import jobs for your user.

> The equivalent API endpoint is `GET /api/importjobs/{jobId}`, where the `jobId` is as returned in the **Response body** in the previous step. There are two links to the right of the import job in the GUI - one enables you to view the log, and the other enables you to download the report file in the `reportFileFormat` specified above.

12. Click **Clear** followed by **Cancel**.
13. Click on the endpoint to collapse it again.

There is also a **SignalR** websocket endpoint available at `jobnotifications` that you can use in order to be notified of job status changes. The endpoint is associated with the following two methods:

- *ReceiveJobStatusChanged*. The event raised by a change in the status, for example when the system starts or finishes processing the job. The update details of the job are included.
- *ReceiveJobLogMessage*. The event raised when the a new log message is added to the job's log. The details of the log message are included.

To connect to the endpoint, you need to include the value of your **API Key** in the header in the same way as when calling the regular API Import endpoints (in accordance with **Authorisation**). You also need to include the email address of the user who created the job, in the connection request.

Importing records from an Excel file

Let us suppose that we wish to import Research Groups from an Excel file as part of an API Import job.

1. Read through **Making an API import request**.
2. From the copy of your import file, remove all the worksheets that you do not need. For this example, we only require the **ResearchGroup** worksheet.

   All worksheets need to specify, on every row in column A (`UKPRN`) the UKPRN associated with the HEI for which records are to be imported. The following is an example of a brief **ResearchGroup** worksheet that we shall import:

UKPRN	**UnitOfAssessment**	**MultipleSubmission (leave blank for this example, so that it defaults to `false`)**	**Code**	**Name**
10007848	34		A	High Culture
10007848	34		B	Low Culture
10007848	34		C	Monobrow

3. Ensure that you have BASE64-encoded your import file and have the results to hand.
4. Go to the UAT system: **https://testsubmissionsapi.ref.ac.uk/swagger/index.html**.
5. Enter the value of your **API Key** in accordance with **Authorisation**.
6. Under **ImportJobs**, click `POST /api/importjobs` to expand the endpoint that allows you to create an import job.
7. Click **Try it out**.
8. Update the *Import Job* model as in the following example:

```
{
   "fileFormat": "ExcelXlsx",
   "fileName": "UOA_34_Research_Groups.xlsx",
   "reportFileFormat": "ExcelXlsx",
   "importMode": "Merge",
   "importWithErrors": false,
   "retrieveCitationCounts": false,
   "zippedFile": false,
   "fileContent":
"UEsDBBQABgAIAAAAIQBBN4LPbgEAAAQFAAATAAgCW0NvbnRlbnRfVHlwZXNdLnhtbCCiBAIooAACAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACsVMluwjAQvVfqP0S+Vomhh6qqCBy6H
Fsk6AeYeJJYJLblGSj8fSdmUVWxCMElUWzPWybzPBit2iZZQkDjbC76WU8kYAunja1y8T39SJ9FgqSs
Vo2zkIs1oBgN7+8G07UHTLjaYi5qIv8iJRY1tAoz58HyTulCq4g/QyW9KuaqAvnY6z3JwlkCSyl1GGI
4eINSLRpK3le8vFEyM1Ykr5tzHVUulPeNKRSxULm0+h9J6srSFKBdsWgZOkMfQGmsAahtMh8MM4YJEL
ExFPIgZ4AGLyPdusq4MgrD2nh8YOtHGLqd4662dV/8O4LRkIxVoE/Vsne5auSPC/OZc/PsNMilrYkty
```

```
lpl7E73Cf54GGV89W8spPMXgc/oIJ4xkPF5vYQIc4YQad0A3rrtEfQcc60C6Anx9FY3F/AX+5QOjtQ4
OI+c2gCXd2EXka469QwEgQzsQ3Jo2PaMHPmr2w7dnaJBH+CW8Q4b/gIAAP//AwBQSwMEFAAGAAgAAAA
hALVVMCP0AAAATAIAAAsACAJfcmVscy8ucmVscyCiBAIooACAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAACskk1PwzAMhu9I/IfI99XdkBBCS3dBSLshVH6ASdwPtY2jJBvdvyccEFQag
wNHf71+/Mrb3TyN6sgh9uI0rIsSFDsjtnethpf6cXUHKiZylkZxrOHEEXbV9dX2mUdKeSh2vY8qq7io
oUvJ3yNG0/FEsRDPLlcaCROlHIYWPZmBWsZNWd5i+K4B1UJT7a2GsLc3oOqTz5t/15am6Q0/iDlM7NK
ZFchzYmfZrnzIbCH1+RpVU2g5abBinnI6InlfZGzA80SbvxP9fC1OnMhSIjQS+DLPR8cloPV/WrQ08c
udecQ3CcOryPDJgosfqN4BAAD//wMAUEsDBBQABgAIAAAAIQAlbbcLbAMAAPoHAAAPAAAAeGwvd29ya
2Jvb2sueG1srFVtb9s2EP4+YP9B0HdFpN4sCXEK25K2AG0XpGn6xUDBSJTFRRI1koodBP3vO9KWkzRF
kaUzbNLkUQ+fu3vudPpu17XWHRWS8X5u4xNkW7QvecX6zdz+fFU4sW1JRfqKtLync/ueSvvd2e+/nW6
5uL3h/NYCgF7O7UapIXVdWTa0I/KED7QHS81FRxQsxcaVg6Ckkg2lqmtdD6HI7Qjr7T1CKl6DweualT
Tj5djRXu1BBG2JAvqyYYOc0LryNXAdEbfj4JS8GwDihrVM3RtQ2+rK9HzTc0FuWnB7h0NrJ+AbwQ8jG
LzpJjC9uKpjpeCS1+oEoN096Rf+Y+Ri/CwEu5cxeB1S4Ap6x3QOj6xE9EZW0REregTD6JfRMEjLaCWF
4L0RLTxy8+yz05q19HovXYsMw0fS6Uy1ttUSqfKKKVrN7Rks+ZY+2xDjsBxZC1YP+15su2dHOV8Iq6I
1GVt1BUKe4KEyoijxQn0ShLFoFRU9UXTFewU6PPj1q5oz2KuGg8KtS/rPyASFwgJ9ga8wkjIlN/KCqM
YaRTu31+uG1iV1lGCdVFzQ9V913bKeFp/WG0Fp/zdff7i3pmKR6yeKJS/L4z9olpQ6EC5EYs92///7q
ABpkU66vFDCgv/n2XvIzSdyB5kCPVSHQj6HVGD/a1+KFH99iDNvtcJ54KBl5jmBX8ROssiRs4xQiBYx
yuMk/AbOiCgtORlVcxCBhp7bAf6B6QPZTRaM0pFVjzQe0OHj6Pm7YbJ90w7rdnfN6FY+ykUvrd0X1ld
8CxfECJy6n5YYepxtbY3xC6tUA04mUXzc+5OyTQOMMfbjUJP2NLO5/RDOlj72Vxl4jJETrIKFk3iL3A
mxHxRosZjNCmQYuU8omcYK1Mxs9aYYLqmkRJTNH4KPAzRy3XtNrG1LpPoqcV5hk8vp6ZK0JdSAnszBB
CMv0SfoTr2XyswgPwYscYAWM5RAmnI/dII48Zw48D1nFWReHs7yLF+aNOn3Q/p/dElTBen04tEsGyLU
lSDlLbyuLmm9JBJ0tXcI+D4luwzjJfKBYlDgwglwAnJaRoETZoUfznC2ysNCa2pPVrtfv7FHxa55mhI
1Qv3q0jXrVI/FYfe4We83Dul6VoLpZabjfnj6ZweLa5PCH97jmjjo0WTPnaJ39i8AAAD//wMAUEsDBB
QABgAIAAAAIQCBPpSX8wAAALoCAAAaAAgBeGwvX3JlbHMvd29ya2Jvb2sueG1sLnJlbHMgogQBKKAAA
QAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAACsUk1LxDAQvQv+hzB3m3YVEdl0LyLsVesPCMm0KdsmITN+9N8bKrpd
WNZLLwNvhnnvzcd29zOO4gMdT9cEqrqIoSBHoTbO87BW/N880DCGLtrR6CRwUTEuzq66vtCw6acxO5PpL
ILJ4UOOb4KCUZh6OmIkT0udKGNGrOMHUyanPQHcpNWd7LtOSA+oRT7K2CtLe3IJopZuX/uUPb9gafgn
kf0fMZCUk8DXkA0ejUISv4wUX2CPK8/GZNec5rwaP6DOUcq0seqjU9fIZ0IIfIRx9/KZJz5aKZu1Xv4
XRC+8opv9vyLMv072bkycfV3wAAAP//AwBQSwMEFAAGAAgAAAAhAIczUrAwAwAAhQgAABgAAAB4bC93
b3Jrc2hlZXRzL3NoZWV0MS54bWykltuO2jAQhu8r9R0s35PEIYGACKtlWdS9qFR1e7g2jgGLJE5tA7u
t+u4dOxACrPZUCchhMt/8Mx5PGF09FDnacjzFZp0EI21omdFcljzFj1zjq/
HHD6OdVGu94twgIJQ6xStjqqHva7biBdWerHgJloVUBTVwqZa+rhSnmXMqcj8Mgp5fUHimjBUr2HIx
UIwPpVsU/DS1BDFc2pAv16JSh9oBXsNrqBqvak6TBYVIOYiF+bRQTEq2PBuWUpF5znk/UAiytCDgk8I
3+4hjLt/EakQTEktF8YDsl9rvkx/4A98yhrSZf6vwpDIV3wr7AIeUeH7JJG4YYVHWPedsF4Ds+VSw43
IUvwnSOKoe0uSTn8WdjtREE86yXQw6dwMpuFgRqKY9MO/eDzKBKywzQopvkjxNReRtgfj1z//BB8p1
vnyND5Pc85MxxiEIxse86lXNsH7+BWAETtHrBEyozY8hue5ymeRNDhv1wMOIUAfhOhfX6INnMN/UWhj
C/oJjdf5e4TF8uVgbAxpGn7ZJg9Trlm0KAQ2AtjS2UyBwT8okLYnQYNRh9qqSIzK+vtRWHcTwg8j9hG
G1n83Fv2/rUnLIzzhOOu8SRR0LN+c67NTFgtzzJgPRwDjgdGzwuTmMRvoUDhHAWOe0q/BXkmBRDqHOF
4CA/ZXDr4ddHcekypoeORkjsE+wSy0xW1U4cMAVIX3+vDqVkJtp7IugJPLEYX+oBZxDUw4CkN19txMP
K3sNhsb5u0beTUdtO2hae2advWPbXdtm1RY/MhoyYtKMJFWlDSN6VlGbZANi0SBEE/iZKz7EJn7B5Fu
IJMwbMpSHwmvm074k7EQy/9t3jLeEG8XcDt+EI8eDbie2fi27bB05W3M+C8od5aect4QXz0tHjwbMT3
z8S3beTYp3Xp61FVb42KLvlnqpai1CjnCzd6+hipejYFHpwbWdmBZPfJXBoYMIerFbyXOWyKwINaLaQ
0hwsYXJZ7z82mQlIJGGnuVZviSiqjqDCw++D+bwmGfFoJKAC82LH9S2EEO7mlhnb+q7uMuCnb/IEY/w
MAAP//AwBQSwMEFAAGAAgAAAAhAMEXEL5OBwAAxiAAABMAAAB4bC90aGVtZS90aGVtZTEueG1s7FnNi
xs3FL8X+j8Mc3f8NeOPJd7gz2yT3SRknZQZdCbz2U0kADDb30jwkk
tOkf0SfN2COt5SSbbEpadg2LR/69p6f3nn5683Tx0r2YekeYC8KSll++UPI9nIzYmCTTln9rOCg0fE9
IlIwRZQlu/Evbn/FYn/wCVRCvKjpX0jbBtKtRKyQIbOyoqZMaQCoadi5tbe/4fOo8XOhUS8HjVpI9+pQ+/
/iXEaPB0gdmVW7 ...
```

```
ulUq0YI5L4XoJiUHt9MiEj7A2VSn97qbxP4TGRQg2MKN9XqrElobHjw7JCiIXoUu4dIdryYZ4xOx7ie
9L3KBISfmj5Jf3nF7cvFtFWJkTlBllDbqD/MrlMYHxY0XPy6cFq0iAIg1p7pV8DqFzH9ev9Wr+20qcB
aDSClaa22DrrlW6QYQ1Q+tWhu1fvVcsW3tBfXbO5HaqPhdegVH+whh8MuuBFC69BKT5cw4edZqdn69e
gFF9bw9dL7V5Qt/RrUERJcriGLoW1ane52hVkwuiOE94Mg0G9kinPUZANq+xSU0xYIjflWozuMj4AgA
JSJEniycUMT9AIsriLKDngxNsl0wgSb4YSJmC4VCkNSlX4rz6B/qYjirYwMqSVXWCJWBtS9nhixMlMt
vwroNU3IC+ePXv+8Onzh789f/To+cNfsrm1KktuByVTU+7Vj1///f0X3l+//vDq8Tfp1CfxwsS//PnL
l7//8Tr1sOLcFS++ffLy6ZMX333150+PHdrbHB2Y8CGJsfCu4WPvJothgQ778QE/ncQwQsSSQBHodqj
uy8gCXlsg6sJ1sO3C2xxYxgW8PL9r2bof8bkkjpmvRrEF3GOMdhh3OuCqmsvw8HCeTN2T87mJu4nQkW
vuLkqsAPfnM6BX4lLZjbBl5g2KEommOMHSU7+xQ4wdq7tDiOXXPTLiTLCJ9O4Qr4OI0yVDcmAlUi60Q
2KIy8JlIITa8s3eba/DqGvVPXxkI2FbIOowfoip5cbLaC5R7FI5RDE1Hb6LZOQycn/BRyauLyREeoop
8/pjLIRL5jqH9RpBvwoM4w77Hl3ENpJLcujSuYsYM5E9dtiNUDxz2kySyMR+Jg4hRZF3g0kXfI/ZO0Q
9QxxQsjHctwm2wv1mIrgF5GqalCeI+mXOHbG8jJm9Hxd0grCLZdo8tti1zYkzOzrzqZXauxhTdIzGGH
u3PnNY0GEzy+e50VciYJUd7EqsK8jOVfWcYAFlkqpr1ilylwgrZffxlG2wZ29xgngWKIkR36T5GkTdS
l045ZxUep2ODk3gNQLlH+SL0ynXBegwkru/SeuNCFlnl3oW7nxdcCt+b7PHYF/ePe2+BBl8ahkg9rf2
zRBRa4I8YYYICgwX3YKIFf5cRJ2rWmzulJvYmzYPAxRGVr0Tk+SNxc+Jsif8d8oedwFzBgWPW/H7lDq
bKGXnRIGzCfcfLGt6aJ7cwHCSrHPWeVVzXtX4//uqZtNePq9lzmuZ81rG9fb1QWqZvHyByibv8uieT7
yx5TMhlO7LBcW7Qnd9BLzRjAcwqNtRuie5agHOIviaNZgs3JQjLeNxJj8nMtqP0AxaQ2XdwJyKTPVUe
DMmoGOkh3UrFZ/QrftO83iPjdNOZ7msupqpCwWS+XgpXI1Dl0qm6Fo9796t1Ot+6FR3WZcGKNnTGGFM
ZhtRdRhRXw5CFF5nhF7ZmVjRdFjRUOqXoVpGceUKMG0VFXjl9uBFveWHQdpBhmYclOdjFae0mbyMrgr
OmUZ6kzOpmQFQYi8zII90U9m6cXlqdWmqvUWkLSOMdLONMNIwghfhLDvNlvtZxrqZh9QyT7liuRtyM+
qNDxFrRSInuIEmJlPQxDtu+bVqCLcqIzRr+RPoGMPXeAa5I9RbF6JTuHYZSZ5u+HdhlhkXsodElDpck
07KBjGRmHuUxC1fLX+VDTTRHKJtK1eAED5a45pAKx+bcRB0O8h4MsEjaYbdGFGeTh+B4VOucP6qxd8d
rCTZHMK9H42PvQM65zcRpFhYLysHjomAi4Ny6s0xgZuwFZHl+XfiYMpo17yK0jmUjiM6i1B2ophknsI
1ia7M0U8rHxhP2ZrBoesuPJiqA/a9T903H9XKcwZp5memxSrq1HST6Yc75A2r8kPUsiqlbv1OLXKuay
65DhLVeUq84dR9iwPBMC2fzDJNWbxOw4qzs1HbtDMsCAxP1Db4bXVGOD3xric/yJ3MWnVALOtKnfj6y
ty81WYHd4E8enB/OKdS6FBCb5cjKPrSG8iUNmCL3JNZjQjfvDknLf9+KWwH3UrYLZQaYb8QVINSoRG2
q4V2GFbL/bBc6nUqD+BgkVFcDtPr+gFcYdBFdmmvx9cu7uPlLc2FEYuLTF/MF7Xh+uK+XNl8ce8RIJ3
7tcqgWW12aoVmtT0oBL1Oo9Ds1jqFXq1b7w163bDRHDzwvSMNDtrVblDrNwq1crdbCGolZX6jWagHlU
o7qLcb/aD9ICtjYOUpfWS+APdqu7b/AQAA//8DAFBLAwQUAAYACAAAACEAF7pfwkYDAABgCAAADQAAA
HhsL3N0eWxlcy54bWykVltv0zAUfkfiP1h+z3JZU9oqCVrXRUKCCWlD4tVNnNbCl8hxRwriv3Nsp00m
KpjGHhafY5/vfOdmN3vfC46eqO6YkjmOryKMqKxUzeQux18ey2CBUWeIrAlXkub4SDv8vnj7JuvMkdO
HPaUGAYTscrw3pl2FYVftqSDdlWqphJ1GaUEMiHoXdq2mpO6skeBhEkXzUBAmsUdYieolIILob4c2qJ
RoiWFbxpk5OiyMRLX6sJNKky0Hqn08IxXq47lOUK9PTpz2Dz+CVVp1qjFXgBuqpmEV/ZPuMlyGpBqRA
Pl1SHEaRsmz2Hv9SqRZqOkTs+XDRdYoaTpUqYM0OU6AqE3B6ptU32Vpt6DCw6ki636gJ8JBE+OwyCrF
lUYGSgeZcxpJBPUnbglnW83ssYYIxo9enViFq/ZwTjDIvVWGlodnc9mP3m1zXJaR+7MWo7MbzQhHXyS
DJqTo08NFr4MD9+nAEeN8ErZXFBn0h6FalrCLhvXjsYX4JLSy5wlb/zy90+QYJ+nEIHQOi2yrdA2jMy
b8pCoyThsDcWm229uvUS383ypjoL2KrGZkpyThNlce5Lkl jBxMV44FrdlBQMl8ec5pm5ebpeNj3QxeX
mzjODlKLzYB+if2L7bxwV6OdQgaSldRzh9ssF+bcx5tj/YNkgdRCvOhzjFcSrajTkso2rD0ufOCzeUU
zWNPYJNXwaK+OeM/IxWPpOzyEinQn6wRaVt+vD+ILdWluxLtmHmtHc1RWrumGuUbznZSUH+kyGAWvYj
2SrMfYGqH2PYBtve4YZWVoWAYMVk7s2vXKn0zJAhSMsn7s6yf84fsQOb43jLlkyi2B8YNkxcyDph1P9
Ywsj6NvYhddc9eoJQ1bciBm8fzZo7H9SfX8XB1Dac+sydlHESOx/VHO1bx3PqgvfnYwQzAFx00y/HPu
/W75eauTIJFtF4Es2uaBst0vQnS2e16symXURLd/po8B//xGLjXC5o1nq06Dk+GHoIdyD+MuhxPBE/f
VQVoT7kvk3l0k8ZRUF5HcTCbk0WwmF+nQZnGyWY+W9+lZTrhnr7y0YjCOPbPjyWfrgwTlDN5qtWpQlM
tFAnEvwQRnioRjj8Nit8AAAD//wMAUEsDBBQABgAIAAAAIQBcB39I6gAAAKMBAAAUAAAAeGwvc2hhcm
VkU3RyaW5ncy54bWx0kN1KAzEQhe8F3yHk3mbrhYhkU+qCCNoqah8g3Z3uBjaTNTOx+vamIApZvJzvn
Dnzo1effhQfEMkFrOVyUUkB2IbOYV/L3dvdxbUUxBY7OwaEWn4ByZU5P9NELHIvUi0H5ulGKWoH8JYW
YQLMyiFEbzmXsVc0RbAdDQDsR3VZVVfKW4dStCEh57lLKRK69wTNLzCanNFsdg/PL1ut2Gh1Aj8QHT8
d1kRA5AG51DdpZDeN8Jr23tHpuNLRhA5KtrV+xtal6XaWVIJ71w+iyRukOIt7DMf/pE3AsI/h+Ben8p
PNNwAAAP//AwBQSwMEFAAGAAgAAAAhADttMkvBAAAAQgEAACMAAAB4bC93b3Jrc2hlZXRzL19yZWxzL
3NoZWV0MS54bWwucmVsc4SPwYrCMBRF9wP+Q3h7k9aFDENTNyK4VecDYvraBtuXkPcU/XuzHGXA5eVw
z+U2m/s0qhtmDoEs1LoChuRCG6iz8Hr/s8qRtmDpEs1LoCheRjF2iw8HvaLb9BsTjq3BQJLTyQYdMuvpoDTk5KiceQWBULsYVRJP0Yw37
E2bGOCamQPubZSYl5MMn5ixvQrKpqbfJfB7QvTrXvLOR9V4M6PVJZ/uyOfR88bqO/zkjyz4RJOZBgPq
JIOchF7fKAYkHrd/aea30OBKZtzMvz9gkAAP//AwBQSwMEFAAGAAgAAAAhAOZfZlbSAAAAsAYAACcAA
AB4bC9wcmludGVyU2V0dGluZ3MvcHJpbnRlclNldHRpbmdzMS5iaW5yZEhhyGdIYkhlUGAIYHBhcGMg
DTCyMLPdYbjCGvy+gZGRgZHhFVc+RwqQ5mfYwMIEpDewMANJH6D5JUCYylBEovnYlDNCBUE0ExDD+Oh
qA4I8wx4pUMFCNCMkwHwBdRBkYABhCNjAsIQZl20wCQMmAYYYb2aGDGcWvA5zc5v/iRWoAqTqPxDi8i
P1fTdq4mAKAVLjfQPQ8cG+IV4gPwgwLBj0kWkAzMCuTi6OoFwM8isyBjk+GFhqJDLkAUupRGDZkTKaO
EdwCIDSBgAAAP//AwBQSwMEFAAGAAgAAAAhAJhUjWNcAQAAbwIAABEACAFkb2NQcm9wcy9jb3JlLnht
bCCiBAEooAABAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAISSX0vDMBTF3wW/Q8l7m7Zb9ye0HUyZIA6EVRTFh5D
crcU2KUl027c3bbfaoeBjcs795ZxL4sWhKp0vULqQIkGB5yMHBJO8ELsEPWUrd4YcbajgtJQCEnQEjR
bp9VXMasKkgkcla1CmAO1YktCE1QnKjakJxprlUFHtWYew4laqihp7VDtcU/ZBd4BD35/gCgzl1FDcA
N26J6ITkrMeWX+qsgVwhqGECoTROPAC/OM1oCr950CrDJxVYY617XSKO2Rz1om9+6CL3rjf7739qI1h
8wf4Zf2waau6hWh2xQClMWeEKaBGqnSTUyWFs7Sd7Zrepn4YvMd4YGiWWVJt1nbv2wL48pjey1w4dwp
AxPi3aultme4J4I6NR7oyZ+V5dHObrVAa+sHc9aduOM+CiIx9Es1fm8cv5pu43UV1ivAvceb6YRaEZB
yRaDIgngFpm/vyi6TfAAAA//8DAFBLAwQUAAYACAAAACEABHwd248BAAAYAwAAEAAIAWRvY1Byb3BzL
2FwcC54bWwgogQBKKAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACckkFv2zAMhe8D9h8M3Rs53VAMgaxiSLf1
sGHBknZnTaZjobIkiIyR7NePttHUWXfajeR7ePpESd0eO1/0kNHFUInlohQFBBtrF/aVeNh9vvogCiQ
TauNjgEqcAMWtfvtGbXJMkMkBFhwRsBItUVpJibaFzuCC5cBKE3NniNu8l7FpnIW7aA8dBJLXZXkj4U
gQaqiv0jlQTImrnv43tI524MPH3SkxsFYfU/LOGuJb6m/O5oixoeLT0YJXci4qptuCPWRHJ10qOW/V1
hoPaw7WjfEISr4M1D2YYWkb4zJq1dOqB0sxF+h+89quRfHLIAw4lehNdiYQYw22qRlrn5Cy/hnzE7YA
hEqyYRqO5dw7r917vRwNXFwah4AJhIVLxJ0jD/i92ZhM/yBezolHhol3wvkBCCbb9kuOh/QKc7w5H/j
XEevYJRNOLJyrry484UPaxTtD8LzVy6HatiZDzQ9x3vp5oO55odkPIevWhD3Uz57XwvAHHqePrpc3i/
Jdyc87myn58qX1HwAAAP//AwBQSwECLQAUAAYACAAAACEAQTeCz24BAAAEBQAAEwAAAAAAAAAAAAAAA
AAAAAAAW0NvbnRlbnRfVHlwZXNdLnhtbFBLAQItABQABgAIAAAAIQC1VTAj9AAAAEwCAAALAAAAAAAA
AAAAAAAAAKcDAABfcmVscy8ucmVsc1BLAQItABQABgAIAAAAIQAlbbcLbAMAAPoHAAAPAAAAAAAAAAAA
AAAAAAMwGAAB4bC93b3JrYm9vay54bWxQSwECLQAUAAYACAAAACEAgT6Ul/MAAAC6AgAAGgAAAAAAAAAA
AAAAAAAABlCgAAeGwvX3JlbHMvd29ya2Jvb2sueG1sLnJlbHNQSwECLQAUAAYACAAAACEAhzNSsDADA
ACFCAAAGAAAAAAAAAAAAAAAAACYDAAAeGwvd29ya3NoZWV0cy9zaGVldDEueG1sUEsBAi0AFAAGAAgA
AAAhAMEXEL5OBwAAxiAAABMAAAAAAAAAAAAAAAAAA/g8AAHhsL3RoZW1lL3RoZW1lMS54bWxQSwECLQA
UAAYACAAAACEAF7pfwkYDAABgCAAADQAAAAAAAAAAAAAAAAB9FwAAeGwvc3R5bGVzLnhtbFBLAQItAB
QABgAIAAAAIQBcB39I6gAAAKMBAAAUAAAAAAAAAAAAAAAAAAO4aAAB4bC9zaGFyZWRTdHJpbmdzLnhtb
FBLAQItABQABgAIAAAAIQA7bTJLwQAAAEIBAAAjAAAAAAAAAAAAAAAAAAocAAB4bC93b3Jrc2hlZXRz
L19yZWxzL3NoZWV0MS54bWwucmVsc1BLAQItABQABgAIAAAAIQDmX2ZW0gAAALAGAAAnAAAAAAAAAAAA
AAAAAAAwdAAB4bC9wcmludGVyU2V0dGluZ3MvcHJpbnRlclNldHRpbmdzMS5iaW5QSwECLQAUAAYACA
AAACEAmFSNY1wBAABvAgAAEQAAAAAAAAAAAAAAAAAjHgAAZG9jUHJvcHMvY29yZS54bWxQSwECLQAUA
AYACAAAACEABHwd248BAAAYAwAAEAAAAAAAAAAAAAAAAAAC2IAAAZG9jUHJvcHMvYXBwLnhtbFBLBQYA
AAAADAAMACYDAAB7IwAAAAA=",
   "tags": "import",
   "outputFilename": "REFExcelRGImport20190802",
}
```

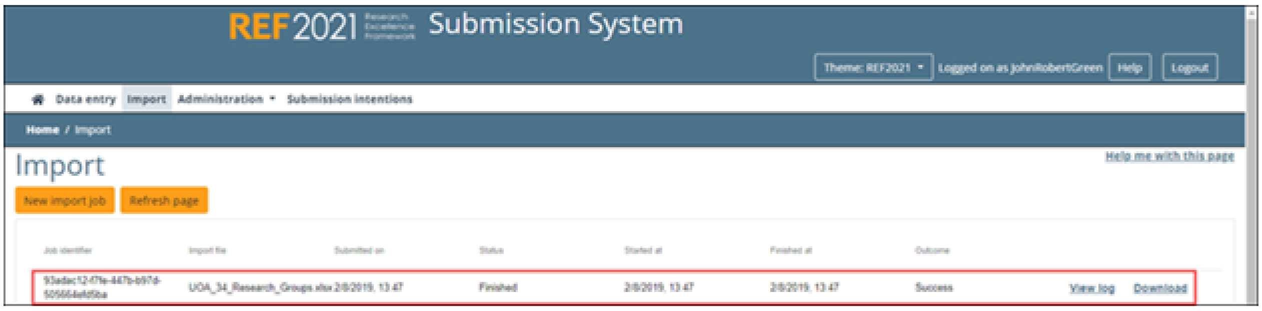
For the parameters, refer to **API import parameters**. Ensure that `fileContent` always holds the BASE64-encoded contents of the file currently being imported.

9. Select the required **Response content type** from the drop-down, for example `application/json`.
10. Click **Execute** to submit the request and show the *cURL* that was submitted.
11. Verify the content returned in the **Response body**, and ensure that `isSuccessful` is set to `true`.

    For example:

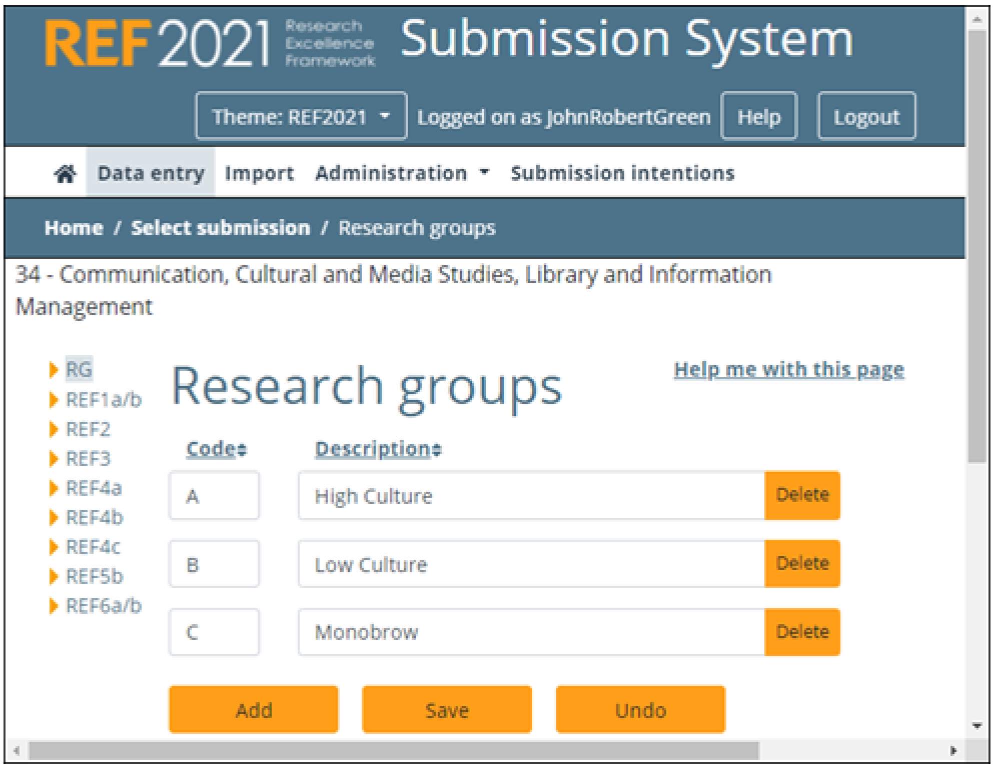
```
{
  "value": {
    "fileFormat": "ExcelXlsx",
    "fileName": "UOA_34_Research_Groups.xlsx",
    "reportFileFormat": "ExcelXlsx",
    "importMode": "Merge",
    "importWithErrors": false,
    "retrieveCitationCounts": false,
    "zippedFile": false,
    "fileContent": null,
    "importFilenameInStore": "UOA_34_Research_Groups-93adac12-f7fe-447b-b97d-
505664efd5ba.xlsx",
    "outputFilenameInStore": "REFExcelRGImport20190802-93adac12-f7fe-447b-b97d-
505664efd5ba",
    "auditLog": null,
    "log": [],
    "jobId": "93adac12-f7fe-447b-b97d-505664efd5ba",
    "tags": "import",
    "status": "Waiting",
    "outcome": "",
    "submittedBy": "JohnRobertGreen",
    "submittedByUserId": "9fccb63f-5c67-4a19-bb33-adcf6471b34f",
    "submittedOn": "2019-07-15T10:29:15.74Z",
    "startedAt": null,
    "finishedAt": null,
    "outputFilename": "REFImport20190715",
    "lastUpdatedByUserId": "9fccb63f-5c67-4a19-bb33-adcf6471b34f",
    "lastUpdatedBy": "JohnRobertGreen",
    "lastUpdatedOn": "2019-08-02T12:47:29.64Z",
    "versionNumber": 1,
    "processingContext": null
  },
  "isSuccessful": true,
  "isFatal": false,
  "errors": []
}
```

12. You can now verify the results in the GUI by looking at the import jobs for your user:



Job identifier	Import file	Submitted on	Status	Started at	Finished at	Outcome		
93adac12-f7fe-447b-b97d-505664efd5ba	UOA_34_Research_Groups.xlsx	2/8/2019, 13:47	Finished	2/8/2019, 13:47	2/8/2019, 13:47	Success	View log	Download

13. The job has finished successfully, so we can now verify the results by looking at the Research Groups in the required UOA:



34 - Communication, Cultural and Media Studies, Library and Information Management

Research groups

Code	Description	
A	High Culture	Delete
B	Low Culture	Delete
C	Monobrow	Delete

14. Click **Clear** followed by **Cancel**.
15. Click on the endpoint to collapse it again.

Importing records from an XML file

Let us suppose that we wish to use a zipped XML file to import Research Groups, research outputs (REF2), and requests to have outputs reduced for certain staff members (REF6a/b).

We will then update the REF6b rationale using a *PUT* statement.

1. Read through **Making an API import request**.
2. Ensure that your import file is syntactically correct and that the UKPRN associated with the HEI for which records are to be imported, is specified within the `<institution>` tags at the top of the file:

```
<ref2021Data xmlns="http://www.ref.ac.uk/schemas/ref2021data">
    <institution>10007848</institution>
        <submissions>
                <submission>
                <unitOfAssessment>11</unitOfAssessment>
                   <researchGroups>
                      <group>
                        <code>A</code>
                        <name>Virtual Private Networks</name>
                      </group>
                      <group>
                        <code>B</code>
                        <name>Firewalls</name>
                      </group>
                     <group>
                        <code>C</code>
                        <name>Application Programming interfaces</name>
                     </group>
                  </researchGroups>
                  <outputs>
                      <output>
                        <outputIdentifier>IOT_Security</outputIdentifier>
                        <outputType>D</outputType>
                        <title>Apparatus: A framework for security analysis in
                               internet of things systems</title>
                        <volumeTitle>Ad Hoc Networks</volumeTitle>
                        <volume>92</volume>
                        <year>2018</year>
                        <issn>1570-8705</issn>
                        <openAccessStatus>Compliant</openAccessStatus>
                        <doi>10.1016/j.adhoc.2018.08.013</doi>
                     </output>
                     <output>
                        <outputIdentifier>Chaos_Encryption</outputIdentifier>
                        <outputType>D</outputType>
                        <title>Fractional chaos based-cryptosystem for generating
                               encryption keys in Ad Hoc networks</title>
                        <volumeTitle>Ad Hoc Networks</volumeTitle>
                        <volume>97</volume>
                        <year>2019</year>
                        <doi>10.1016/j.adhoc.2019.102005</doi>
                    </output>
                  </outputs>
                  <unitCircumstances>
                     <removeMinimumOfOneRequests>
                        <request>
                        <hesaStaffIdentifier>0000983494899</hesaStaffIdentifier>
                        <circumstances>
                            <circumstance>ECR</circumstance>
                            <circumstance>FamilyRelatedLeave</circumstance>
                        </circumstances>
                        <supportingStatement>Supporting statement.
                        </supportingStatement>
                        </request>
                        <request>
                        <hesaStaffIdentifier>0000911241489
                        </hesaStaffIdentifier>
                        <circumstances>
                            <circumstance>SecondmentsOrCareerBreaks</circumstance>
                            <circumstance>ECR</circumstance>
                        </circumstances>
                        <supportingStatement>Another supporting statement.
                        </supportingStatement>
                        </request>
                     </removeMinimumOfOneRequests>
                        <circumstances>
                            <circumstance>
                            <hesaStaffIdentifier>0000911241489
                            </hesaStaffIdentifier>
                            <typeofCircumstance>SecondmentsOrCareerBreaks
                            </typeofCircumstance>
                            <tariffBand>3</tariffBand>
                            </circumstance>
                        </circumstances>
                     <unitRationaleStatement>Unit rationale statement.
                     </unitRationaleStatement>
                   </unitCircumstances>
                </submission>
        </submissions>
</ref2021Data>
```

3. Ensure that you have BASE64-encoded your import file and have the results to hand.
4. Go to the UAT system: https://testsubmissionsapi.ref.ac.uk/swagger/index.html.
5. Enter the value of your **API Key** in accordance with Authorisation.
6. Under **ImportJobs**, click `POST /api/importjobs` to expand the endpoint that allows you to create an import job.
7. Click **Try it out**.
8. Update the *Import Job* model as in the following example:

```
{
        "fileFormat": "Xml",
        "fileName": "CHESTER_RG_Ref6a_b_Rationale.zip",
        "reportFileFormat": "ExcelXlsx",
        "importMode": "Merge",
        "zippedFile": true,
        "fileContent":
"UEsDBBQAAgAIADFOPk9mwxUbCAMAAOcJAAAgAAAAQ0hFU1RFUl9SR19SZWY2YV9iX1JhdGlvbmFsZS5
4bWytVmFv0zAQ/QwS/8Hi++qk61g7mUhdt8EkoGgbfJ285LKaJXbwOS3995ybtMmURkysVqTkPZ/P9nu
nU4SFdBgMwwvpJPuTZxo/vl84V5xxvlqtBjQ7kPGgfOIYLyCXyOv4hOLfR+/evhFKo1OudMroKAyC4HQ
8GgveZn0Ulg+5QiSIHreJCpdauXk6RQTEHLSLwlDwDkmhFGsBQdp48cmasqjyEfvoUQUIxSaBaCq4f+9
ILXOIfirrSpmx71YtpQP2DdzK2CcU3E9vs/Eqnf/uz3++L/+VsrCSWbY/YX+y2b5k06LIVCy9jnRg82h
lniv9yJR2YFMZQ/8uBDpCsdYQpnRF6bb61dCjFrxOSHaVKrDR9fzu/hbi0iq3Frwz/Xzd3bqA6ELwBu0
CnHLZ5mLSSlfiGZuylK4F3gSWGsuw3oRJLbM1KqTbVhfW4JhJmVuQBMhwjQ5yEsBnbPIvTVbmcFftkrD
PJm5ZvJ3sLogmw3q6mVmTetEwCKmg/eeOV4hU7CenwdH4NDgR3ONGgAL0NCZn8Nb5C0Yzk5OJUjvBO3P
bRYlRURgMwiD8wH8NZLIw8cBvPAjoCY8Fp4A6eCvqS32bLaTB+0sd23Xh6+j13l1ZGftMMmOxT84eJEJ
ytNnAVK5sjHwEDeSxr1fYbc+eYL1xtPZGV94czsbTXhsnbRv7RZ8QMwyCkz7RG4RR1R7q/jVTNi5zdFK
Tw9uFFnKzhK9Kq7zM5+lcww38LgFdY76tiBoTsQCUVCBp2vIooDEZH48mo/GELrInpFkfdw7SoaPL2Y3
gbaYn7krmKlvfQCYdJF9ALqFvGe/ZFsuiMNaXgS962DTy2x3HcEsOBO+GNoXNa5n+Q7YwHI7C0fgQslE
HNDrxB8O5nUkLYM8tyCfsqNIj+iHUm2rjFmAZvlpFwv+oz64srDvEnjsRfQBPHLUfk85eaEE3upVJWpW
m51InEfXTBjXHZR1veqxhe8amA9zIqi9CI/wPopnd8s986q5orPKH2d9TBH/+79TGPkLw1j+dD6DnL1B
LAQIUABQAAgAIADFOPk9mwxUbCAMAAOcJAAAgACQAAAAAAAAAIAAAAAAAAABDSEVTVEVSX1JHX1JlZjZ
hX2JfUmF0aW9uYWxlLnhtbAoAIAAAAAAAAQAYACDgrftrd9UBcDKs+2t31QFwMqz7a3fVAVBLBQYAAAA
AAQABAHIAAABGAwAAAAA=",
        "outputFilename": "REFImport20190930"
}
```

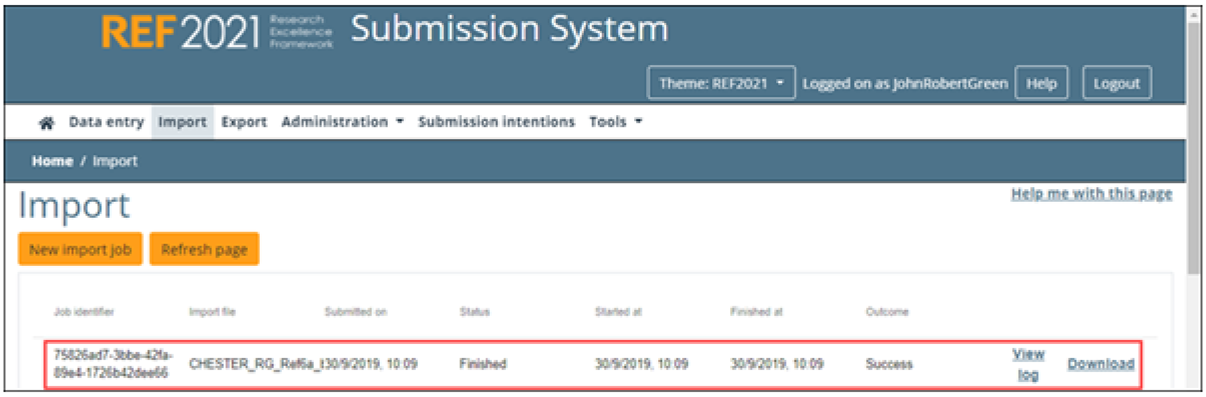
For the parameters, refer to **API import parameters**. Ensure that `fileContent` always holds the BASE64-encoded contents of the file currently being imported.

9. Select the required **Response content type** from the drop-down, for example `application/json`.
10. Click **Execute** to submit the request and show the *cURL* that was submitted.
11. Verify the content returned in the **Response body**, and ensure that `isSuccessful` is set to `true`.

    For example:

```
{
    "value": {
        "fileFormat": "Xml",
        "fileName": "CHESTER_RG_Ref6a_b_Rationale.zip",
        "reportFileFormat": "ExcelXlsx",
        "importMode": "Merge",
        "importWithErrors": false,
        "retrieveCitationCounts": false,
        "zippedFile": true,
        "fileContent": null,
        "importFilenameInStore": "CHESTER_RG_Ref6a_b_Rationale-75826ad7-3bbe-42fa
           -89e4-1726b42dee66.zip",
        "outputFilenameInStore": "REFImport20190930-75826ad7-3bbe-42fa
           -89e4-1726b42dee66",
        "auditLog": null,
        "log": [],
        "jobId": "75826ad7-3bbe-42fa-89e4-1726b42dee66",
        "tags": "import",
        "status": "Waiting",
        "outcome": "",
        "submittedBy": "JohnRobertGreen",
        "submittedByUserId": "9fccb63f-5c67-4a19-bb33-adcf6471b34f",
        "submittedOn": "2019-09-30T09:09:22.533Z",
        "startedAt": null,
        "finishedAt": null,
        "outputFilename": "REFImport20190930",
        "lastUpdatedByUserId": "9fccb63f-5c67-4a19-bb33-adcf6471b34f",
        "lastUpdatedBy": "JohnRobertGreen",
        "lastUpdatedOn": "2019-09-30T09:09:22.397Z",
        "versionNumber": 1,
        "processingContext": null
    },
    "isSuccessful": true,
    "isFatal": false,
    "errors": []
}
```

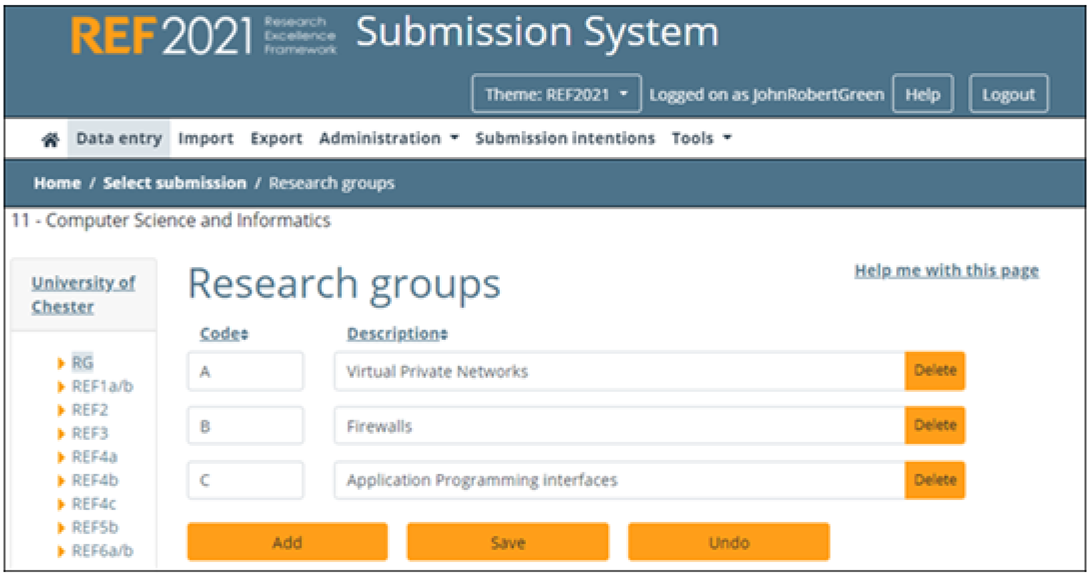
12. You can now verify the results in the GUI by looking at the import jobs for your user:



Import

Job identifier	Import file	Submitted on	Status	Started at	Finished at	Outcome		
75826ad7-3bbe-42fa-89e4-1726b42dee66	CHESTER_RG_Ref6a_t	30/9/2019, 10:09	Finished	30/9/2019, 10:09	30/9/2019, 10:09	Success	View log	Download

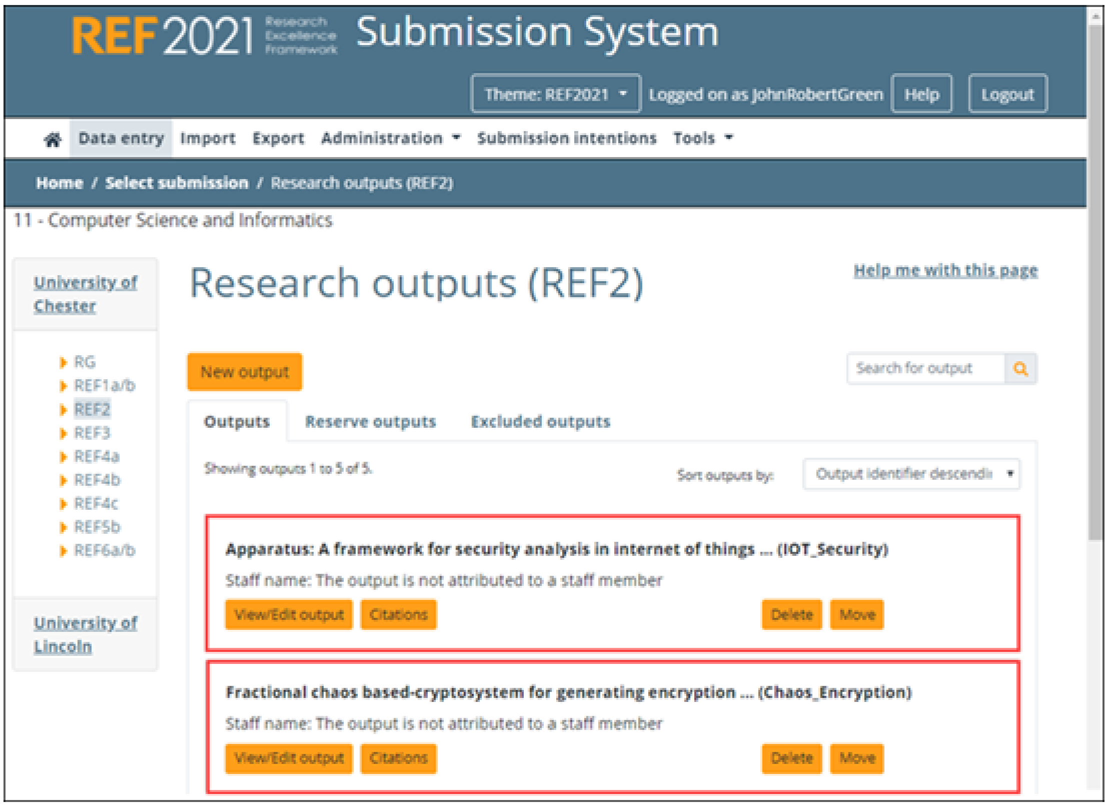
13. The job has finished successfully, so we can now verify the results by looking at the Research Groups, REF2 research outputs, and REF6a/b reduction requests, within the required UOA:



11 - Computer Science and Informatics

Research groups

Code	Description	
A	Virtual Private Networks	Delete
B	Firewalls	Delete
C	Application Programming interfaces	Delete


REF2021 Research Excellence Framework Submission System

Theme: REF2021 | Logged on as JohnRobertGreen | Help | Logout

Data entry | Import | Export | Administration | Submission intentions | Tools

Home / Select submission / Research outputs (REF2)

11 - Computer Science and Informatics

University of Chester
- RG
- REF1a/b
- REF2
- REF3
- REF4a
- REF4b
- REF4c
- REF5b
- REF6a/b

University of Lincoln

Research outputs (REF2)

Help me with this page

New output | Search for output

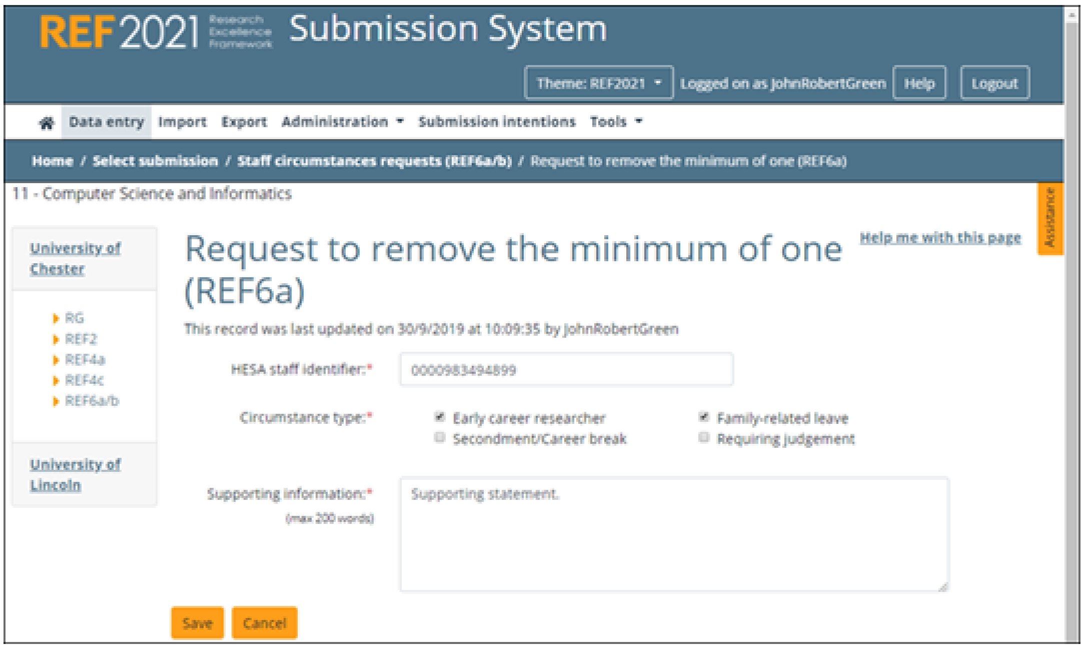
Outputs | Reserve outputs | Excluded outputs

Showing outputs 1 to 5 of 5.

Sort outputs by: Output identifier descendi

Apparatus: A framework for security analysis in internet of things ... (IOT_Security)
Staff name: The output is not attributed to a staff member
View/Edit output | Citations | Delete | Move

Fractional chaos based-cryptosystem for generating encryption ... (Chaos_Encryption)
Staff name: The output is not attributed to a staff member
View/Edit output | Citations | Delete | Move


REF2021 Research Excellence Framework Submission System

Theme: REF2021 | Logged on as JohnRobertGreen | Help | Logout

Data entry | Import | Export | Administration | Submission intentions | Tools

Home / Select submission / Staff circumstances requests (REF6a/b) / Request to remove the minimum of one (REF6a)

11 - Computer Science and Informatics

University of Chester
- RG
- REF2
- REF4a
- REF4c
- REF6a/b

University of Lincoln

Request to remove the minimum of one (REF6a)

Help me with this page

Assistance

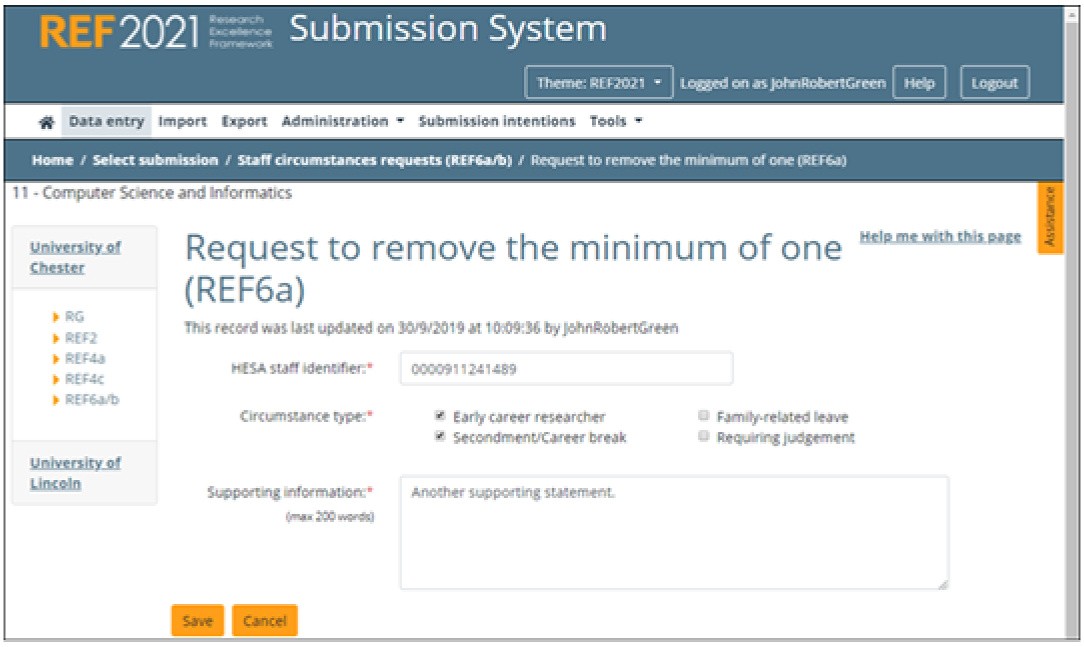
This record was last updated on 30/9/2019 at 10:09:35 by JohnRobertGreen

HESA staff identifier:* 0000983494899

Circumstance type:* [✓ Early career researcher] [✓ Family-related leave] Secondment/Career break, Requiring judgement

Supporting information:* (max 200 words) Supporting statement.

Save | Cancel

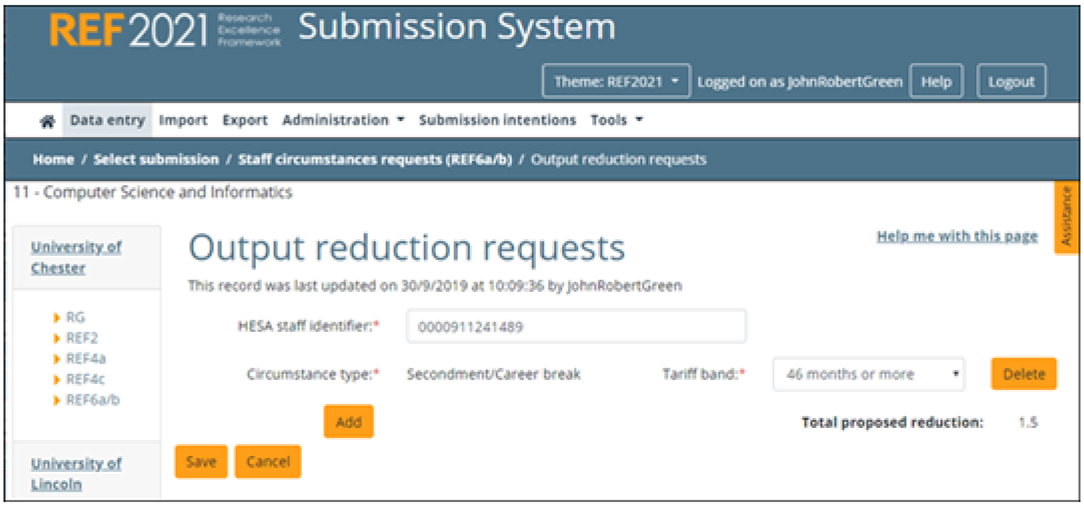

REF2021 Research Excellence Framework Submission System
Theme: REF2021
Logged on as JohnRobertGreen
Help
Logout
Data entry
Import
Export
Administration
Submission intentions
Tools
Home / Select submission / Staff circumstances requests (REF6a/b) / Request to remove the minimum of one (REF6a)
University of Chester
RG
REF2
REF4a
REF4c
REF6a/b
University of Lincoln
Help me with this page
Assistance
Save
Cancel

11 - Computer Science and Informatics

Request to remove the minimum of one (REF6a)

This record was last updated on 30/9/2019 at 10:09:36 by JohnRobertGreen

HESA staff identifier:*	0000911241489	
Circumstance type:*	☑ Early career researcher ☑ Secondment/Career break	☐ Family-related leave ☐ Requiring judgement
Supporting information:* (max 200 words)	Another supporting statement.	

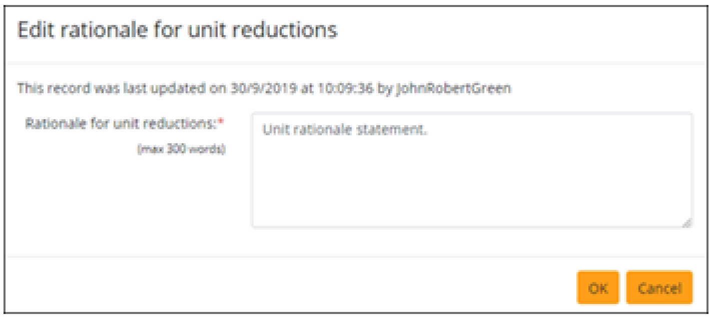


11 - Computer Science and Informatics

Output reduction requests

This record was last updated on 30/9/2019 at 10:09:36 by JohnRobertGreen

HESA staff identifier:*	0000911241489		
Circumstance type:*	Secondment/Career break	Tariff band:*	46 months or more
		Total proposed reduction:	1.5



Edit rationale for unit reductions

This record was last updated on 30/9/2019 at 10:09:36 by JohnRobertGreen

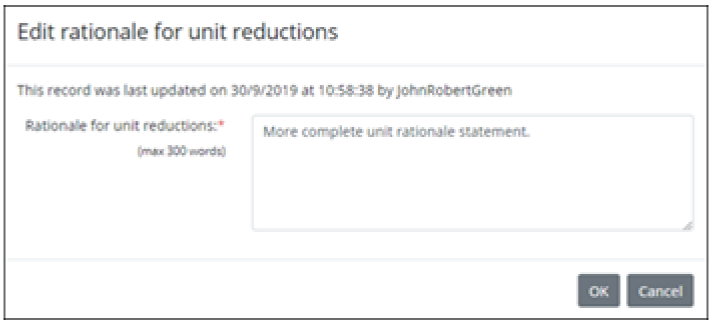
Rationale for unit reductions:* (max 300 words)	Unit rationale statement.

14. We will now update the rationale by using the relevant `PUT` endpoint. However, this endpoint requires the `submissionId`, which can be found as follows:
    - Under **Submissions**, click `GET /api/submissions` to expand the endpoint that allows you to find the required `submissionId`.
    - Click **Try it out**.
    - Click **Execute**.
    - Make a note of the `submissionID` for UOA 11, which, in this case, is `d68cdfe0-13a7-4203-8c9b-1877bb8e3705`.

15. Under **UnitRationaleStatements** in Swagger UI, expand the endpoint `PUT /api/submissions/{submissionId}/circumstances/unitrationalestatement`, and click **Try it out**.
16. Enter the `submissionID`.
17. Update the *Import Job* model as in the following example:

```
{
        "submissionId": "d68cdfe0-13a7-4203-8c9b-1877bb8e3705",
        "statement": "More complete unit rationale statement."
}
```

18. Click **Execute**.
19. Provided that the content is returned successfully, the rationale should now appear as follows in REF6b:



Edit rationale for unit reductions

This record was last updated on 30/9/2019 at 10:58:38 by JohnRobertGreen

Rationale for unit reductions:* (max 300 words)

More complete unit rationale statement.

20. Click **Clear** followed by **Cancel** for all endpoints.
21. Click on them to collapse them again.

Importing records from a JSON file

Let us suppose that we wish to import Research Groups from a JSON file as part of an API Import job.

1. Read through **Making an API import request**.
2. Ensure that your import file is syntactically correct and that the UKPRN associated with the HEI for which records are to be imported, is specified within the `"institution"` object at the top of the file:

```
{
        "$schema": "./ref2021schema.json",
        "institution": "10007848",
        "submissions": [
            {
                "unitOfAssessment": 24,
                "researchGroups": [
                        {
                        "code": "A",
                        "name": "Aerobics"
                        },
                        {
                        "code": "P",
                        "name": "Pilates"
                        },
                        {
                        "code": "Y",
                        "name": "Yoga"
                        }
                ]
            }
      ]
}
```

3. Ensure that you have BASE64-encoded your import file and have the results to hand.
4. Go to the UAT system: **https://testsubmissionsapi.ref.ac.uk/swagger/index.html**.
5. Enter the value of your **API Key** in accordance with **Authorisation**.
6. Under **ImportJobs**, click `POST /api/importjobs` to expand the endpoint that allows you to create an import job.
7. Click **Try it out**.
8. Update the *Import Job* model as in the following example:

```
{
   "fileFormat": "JSON",
   "fileName": "RG_JSON_ref2021example.json",
   "reportFileFormat": "JSON",
   "importMode": "Merge",
   "importWithErrors": false,
   "retrieveCitationCounts": false,
   "zippedFile": false,
   "fileContent":
"ew0KICAiJHNjaGVtYSI6ICIuL3JlZjIwMjFzY2hlbWEuanNvbiIsDQogICJpbnN0aXR1dGlvbiI6IC
IxMDAwNzg0OCIsDQogICJzdWJtaXNzaW9ucyI6IFsNCiAgICB7DQogICAgICAidW5pdE9mQXNzZXNzb
WVudCI6IDI0LA0KICAgICAgInJlc2VhcmNoR3JvdXBzIjogWw0KICAgICAgICB7DQoJCQkiY29kZSI6
ICJBIiwNCgkJCSJuYW1lIjogIkFlcm9iaWNzIg0KCQkJfSwNCgkJCXsNCgkJCSJjb2RlIjogIlAiLA0
KCQkJIm5hbWUiOiAiUGlsYXRlcyINCgkJCX0sDQoJCQl7DQoJCQkiY29kZSI6ICJZIiwNCgkJCSJuYW
1lIjogIllvZ2EiDQoJCQl9DQogICAgICBdDQogICAgfQ0KICBdDQp9",
   "tags": "import",
   "outputFilename": "REFImport_B_20190805",
}
```

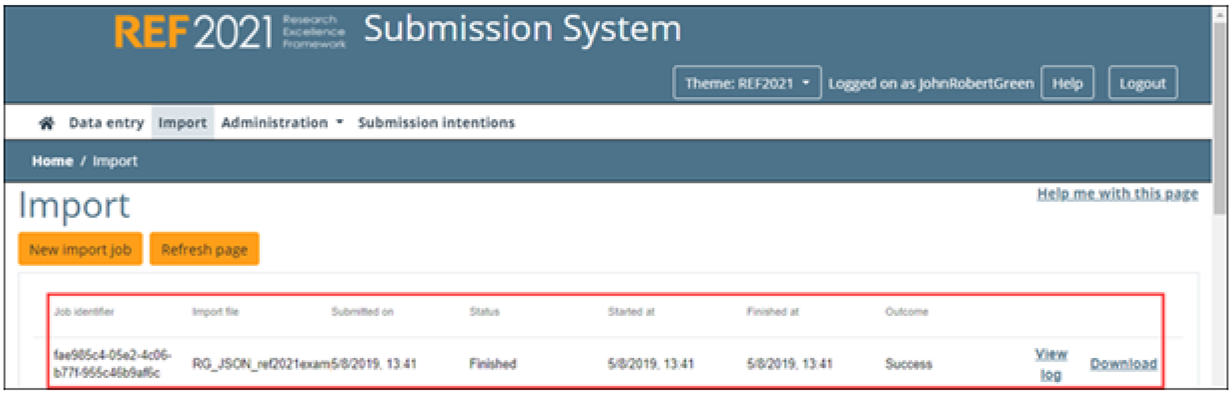
For the parameters, refer to **API import parameters**. Ensure that `fileContent` always holds the BASE64-encoded contents of the file currently being imported.

9. Select the required **Response content type** from the drop-down, for example `application/json`.
10. Click **Execute** to submit the request and show the *cURL* that was submitted.
11. Verify the content returned in the **Response body**, and ensure that `isSuccessful` is set to `true`.

    For example:

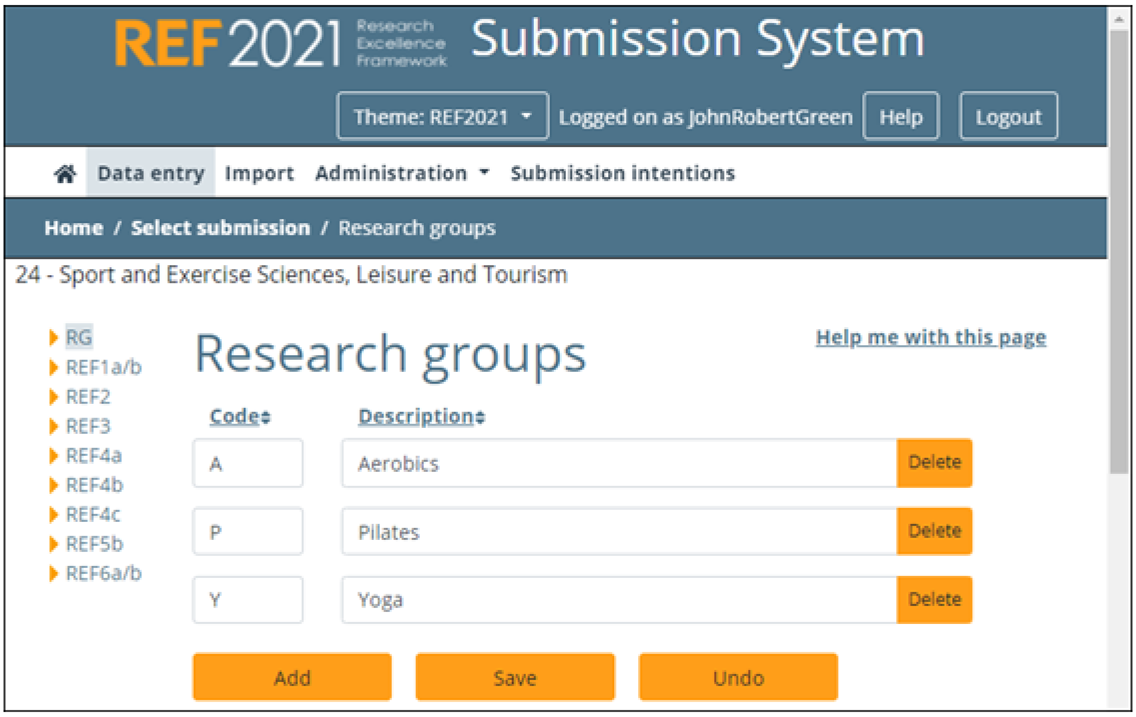
```
{
  "value": {
    "fileFormat": "JSON",
    "fileName": "RG_JSON_ref2021example.json",
    "reportFileFormat": "JSON",
    "importMode": "Merge",
    "importWithErrors": false,
    "retrieveCitationCounts": false,
    "zippedFile": false,
    "fileContent": null,
    "importFilenameInStore": "RG_JSON_ref2021example-fae985c4-05e2-4c06-b77f-
955c46b9af6c.json",
    "outputFilenameInStore": "REFImport_B_20190805-fae985c4-05e2-4c06-b77f-
955c46b9af6c",
    "auditLog": null,
    "log": [],
    "jobId": "fae985c4-05e2-4c06-b77f-955c46b9af6c",
    "tags": "import",
    "status": "Waiting",
    "outcome": "",
    "submittedBy": "JohnRobertGreen",
    "submittedByUserId": "9fccb63f-5c67-4a19-bb33-adcf6471b34f",
    "submittedOn": "2019-08-05T12:41:41.167Z",
    "startedAt": null,
    "finishedAt": null,
    "outputFilename": "REFImport_B_20190805",
    "lastUpdatedByUserId": "9fccb63f-5c67-4a19-bb33-adcf6471b34f",
    "lastUpdatedBy": "JohnRobertGreen",
    "lastUpdatedOn": "2019-08-05T12:41:40.98Z",
    "versionNumber": 1,
    "processingContext": null
  },
  "isSuccessful": true,
  "isFatal": false,
  "errors": []
}
```

12. You can now verify the results in the GUI by looking at the import jobs for your user:



Job identifier	Import file	Submitted on	Status	Started at	Finished at	Outcome		
fae985c4-05e2-4c06-b77f-955c46b9af6c	RG_JSON_ref2021exam	5/8/2019, 13:41	Finished	5/8/2019, 13:41	5/8/2019, 13:41	Success	View log	Download

13. The job has finished successfully, so we can now verify the results by looking at the Research Groups in the required UOA:



Code	Description	
A	Aerobics	Delete
P	Pilates	Delete
Y	Yoga	Delete

14. Click **Clear** followed by **Cancel**.

15. Click on the endpoint to collapse it again.

Replacing previously imported records

Let us suppose that we wish to replace some Research Groups that we previously imported as part of an API Import job. We previously used the **Merge** *importMode* (refer to importMode): we now need to use the **Replace** *importMode* instead.

1. Read through Making an API import request.
2. Ensure that your import file is syntactically correct, referring as required to Importing records from an XML file, Importing records from a JSON file, or Importing records from an Excel file. We will use a JSON file in our example, but the basic procedure is the same, regardless of which format is being used.

   Let us suppose that the Research Groups that we previously imported into UOA 24 using Importing records from a JSON file, need to be replaced by those in the JSON file shown below:

```
{
       "$schema": "./ref2021schema.json",
       "institution": "10007848",
       "submissions": [
           {
               "unitOfAssessment": 24,
               "researchGroups": [
                       {
                       "code": "A",
                       "name": "Aqua Aerobics"
                       },
                       {
                       "code": "H",
                       "name": "Hot Yoga"
                       }
               ]
           }
      ]
}
```

3. Ensure that you have BASE64-encoded your import file and have the results to hand.
4. Go to the UAT system: https://testsubmissionsapi.ref.ac.uk/swagger/index.html.
5. Enter the value of your **API Key** in accordance with Authorisation.
6. Under **ImportJobs**, click `POST /api/importjobs` to expand the endpoint that allows you to create an import job.
7. Click **Try it out**.
8. Update the *Import Job* model as in the following example, taking care to specify **Replace** as the *importMode*:

```
{
   "fileFormat": "JSON",
   "fileName": "RG_JSON_ref2021example_Rep.json",
   "reportFileFormat": "JSON",
   "importMode": "Replace",
   "importWithErrors": false,
   "retrieveCitationCounts": false,
   "zippedFile": false,
   "fileContent":
"ew0KICAiJHNjaGVtYSI6ICIuL3JlZjIwMjFzY2hlbWEuanNvbiIsDQogICJpbnN0aXR1dGlvbiI6IC
IxMDAwNzg0OCIsDQogICJzdWJtaXNzaW9ucyI6IFsNCiAgICB7DQogICAgICAidW5pdE9mQXNzZXNzb
WVudCI6IDI0LA0KICAgICAgInJlc2VhcmNoR3JvdXBzIjogWw0KICAgICAgICB7DQoJCQkiY29kZSI6
ICJBIiwNCgkJCSJuYW1lIjogIkFxdWEgQWVyb2JpY3MiDQoJCQl9LA0KCQkJKA0KCQkJImNvZGUiOiA
iSCIsDQoJCQkibmFtZSI6ICJIb3QgWW9nYSINCgkJCX0NCiAgICAgIF0NCiAgICB9DQogIF0NCn0=",
    "tags": "replace;import",
    "outputFilename": "REFImport_R_20190805",
}
```

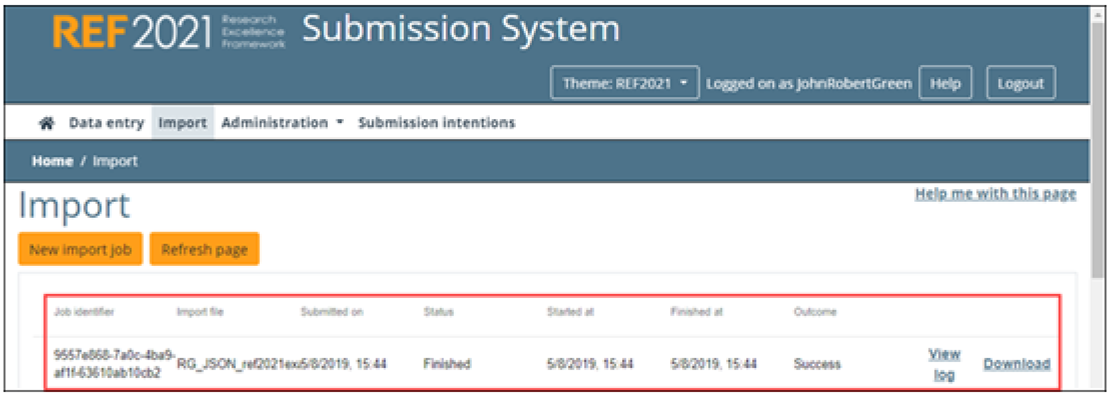
For the parameters, refer to **API import parameters**. Ensure that `fileContent` always holds the BASE64-encoded contents of the file currently being imported.

9. Select the required **Response content type** from the drop-down, for example `application/json`.
10. Click **Execute** to submit the request and show the *cURL* that was submitted.
11. Verify the content returned in the **Response body**, and ensure that `isSuccessful` is set to `true`.

    For example:

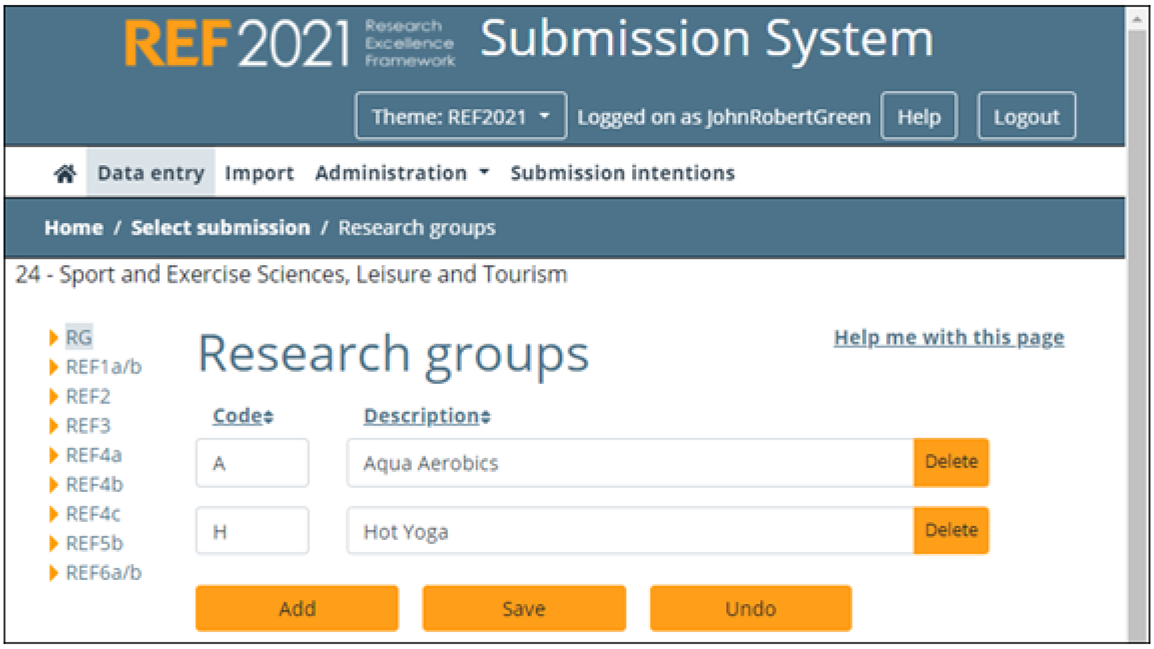
```
{
  "value": {
    "fileFormat": "JSON",
    "fileName": "RG_JSON_ref2021example_Rep.json",
    "reportFileFormat": "JSON",
    "importMode": "Replace",
    "importWithErrors": false,
    "retrieveCitationCounts": false,
    "zippedFile": false,
    "fileContent": null,
    "importFilenameInStore": "RG_JSON_ref2021example_Rep-6a453e78-02f1-4728-
8b5c-c742a760cc73.json",
    "outputFilenameInStore": "REFImport_R_20190805-6a453e78-02f1-4728-8b5c-
c742a760cc73",
    "auditLog": null,
    "log": [],
    "jobId": "9557e868-7a0c-4ba9-af1f-63610ab10cb2",
    "tags": "replace;import",
    "status": "Waiting",
    "outcome": "",
    "submittedBy": "JohnRobertGreen",
    "submittedByUserId": "9fccb63f-5c67-4a19-bb33-adcf6471b34f",
    "submittedOn": "2019-08-05T14:44:32.263Z",
    "startedAt": null,
    "finishedAt": null,
    "outputFilename": "REFImport_B_20190805",
    "lastUpdatedByUserId": "9fccb63f-5c67-4a19-bb33-adcf6471b34f",
    "lastUpdatedBy": "JohnRobertGreen",
    "lastUpdatedOn": "2019-08-05T14:44:32.153Z",
    "versionNumber": 1,
    "processingContext": null
  },
  "isSuccessful": true,
  "isFatal": false,
  "errors": []
}
```

12. You can now verify the results in the GUI by looking at the import jobs for your user:



Job identifier	Import file	Submitted on	Status	Started at	Finished at	Outcome		
9557e868-7a0c-4ba9-af1f-63610ab10cb2	RG_JSON_ref2021ex	5/8/2019, 15:44	Finished	5/8/2019, 15:44	5/8/2019, 15:44	Success	View log	Download

13. The job has finished successfully, so we can now verify the results by looking at the Research Groups in the required UOA:


REF2021 Research Excellence Framework Submission System
Theme: REF2021
Logged on as JohnRobertGreen
Help
Logout
Data entry
Import
Administration
Submission intentions
Home / Select submission / Research groups
24 - Sport and Exercise Sciences, Leisure and Tourism
RG
REF1a/b
REF2
REF3
REF4a
REF4b
REF4c
REF5b
REF6a/b
Research groups
Help me with this page

Code	Description	
A	Aqua Aerobics	Delete
H	Hot Yoga	Delete

As part of the **Replace** action, you will see the following when you refer back to **Importing records from a JSON file**: any existing Research Groups have been overwritten (**A** in this case), any Research Groups that do not exist in the new file have been removed (**P** and **Y** in this case), and any new Research Groups have been added (**H** in this case).

14. Click **Clear** followed by **Cancel**.
15. Click on the endpoint to collapse it again.

Making an API export request

You can replicate the GUI export process using the API endpoint `POST /api/exportjobs`.

Guidance on the process, together with schemas and examples in the different formats, are available from the **Submission system data requirements** web page.

The basic steps are the same for each file format, that is to say:

1. Go to the UAT system: **https://testsubmissionsapi.ref.ac.uk/swagger/index.html**.
2. Enter the value of your **API Key** in accordance with **Authorisation**.
3. Under **ExportJobs**, click `POST /api/exportjobs` to expand the endpoint that allows you to create an export job.
4. Click **Try it out**.
5. Update the *Export Job* model in accordance with the required example from the following sub-topics:
   - **Exporting records to an Excel file**

   The model contains the following parameters, which correspond to the fields described in the GUI Export process:

API export parameters

API Parameter	Equivalent GUI Field	Possible API values (case-insensitive)
`sourceFilters`		The values returned by lookup `GET /api/exportfilters`. Specify one set of `sourceFilters` for each submission for which data is to be exported. Each set of `sourceFilters` consists of the following: • `sourceName` - This is the names of one of the REF forms ("ResearchGroup", "REF1a", "REF1b", "REF2", and so on). • `sourceParameters` - The relevant `submissionId` from those returned by `GET /api/submissions`.
`fileFormat`	**File format**	The format of the file being exported, that is to say: `ExcelXlsx`, `XML` or `JSON`.
`outputFileName`	**File name**	Supply the prefix of the export file name. All export files are zipped, and so will have the `.zip` suffix added automatically.
`tags`	**N/A**	The `tags` parameter is mandatory for exporting, and, if you omit it, then `"tags": "export;forms"` will be supplied by default. If you enter your own `tags`, then you should separate them by semi-colons, but at least one of them should be `"export"`. The `"export"` tag is necessary in order for the associated job to appear in the user's list of export jobs, and for the submission system to be updated.

In order to create an export job, you do not need to include all the endpoint parameters listed in the **Swagger UI** documentation, only those listed above. For example, the *auditLog* and *log* properties are not required as the API will add them automatically.

8. Select the required **Response content type** from the drop-down, for example `application/json`.
9. Click **Execute** to submit the request and show the *cURL* that was submitted.
10. Verify the content returned in the **Response body**, and ensure that `isSuccessful` is set to `true`.

    The user and update information is modified automatically in accordance with the user executing the API.

11. You can now verify the results in the GUI by looking at the export jobs for your user.

    The equivalent API endpoint is `GET /api/exportjobs/{jobId}`, where the `jobId` is as returned in the **Response body** in the previous step. There are two links to the right of the export job in the GUI - one enables you to download the zipped export file, and the other enables you to view the log. (The equivalent API endpoints are, respectively: `GET /api/exportjobs/{jobId}/outputfile` and `GET /api/exportjobs/{jobId}/log`.)

12. Click **Clear** followed by **Cancel**.
13. Click on the endpoint to collapse it again.

There is also a **SignalR** websocket endpoint available at `jobnotifications` that you can use in order to be notified of job status changes. The endpoint is associated with the following two methods:

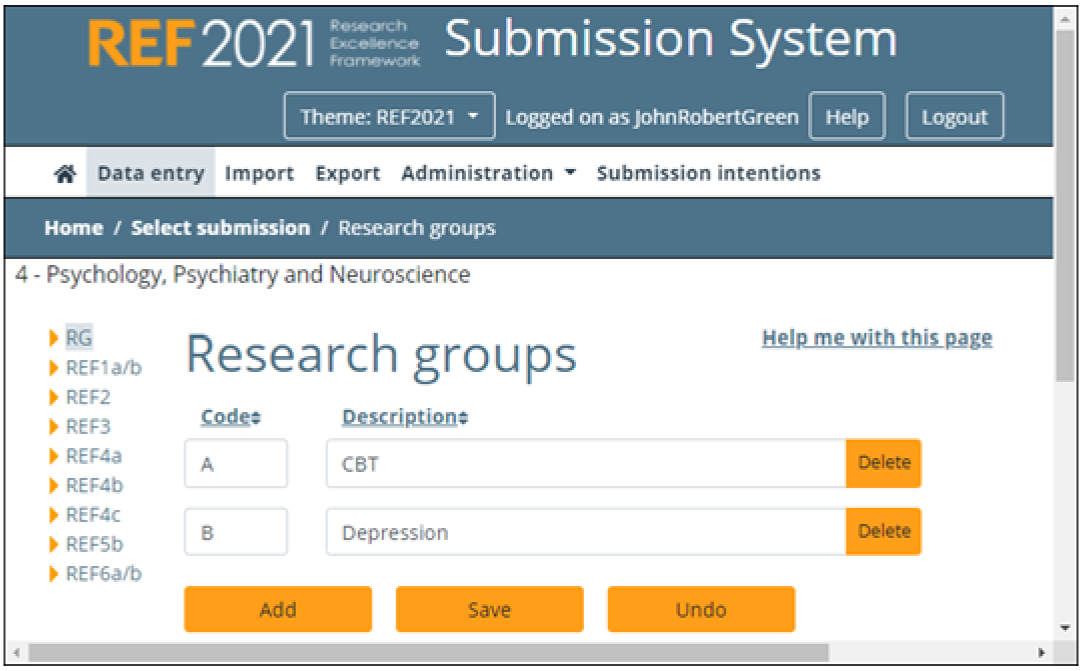
- *ReceiveJobStatusChanged*. The event raised by a change in the status, for example when the system starts or finishes processing the job. The update details of the job are included.
- *ReceiveJobLogMessage*. The event raised when the a new log message is added to the job's log. The details of the log message are included.

To connect to the endpoint, you need to include the value of your **API Key** in the header in the same way as when calling the regular API Export endpoints (in accordance with **Authorisation**). You also need to include the email address of the user who created the job, in the connection request.

Exporting records to an Excel file

Let us suppose that we wish to export Research Groups to an Excel file.

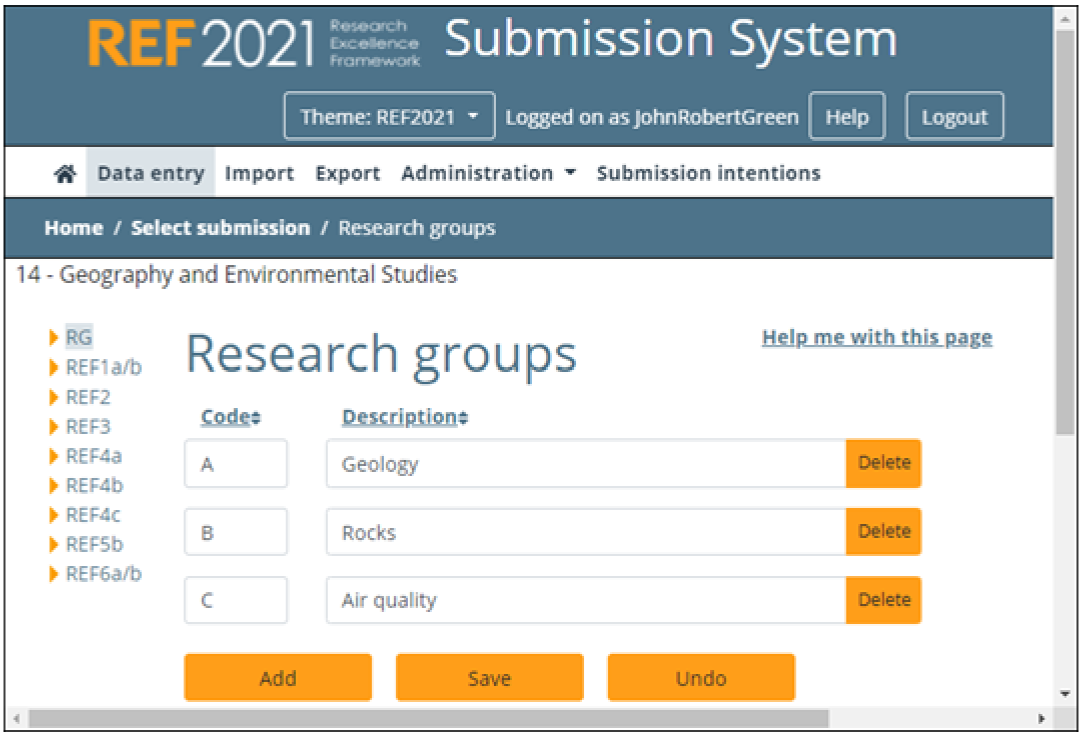
1. Read through **Making an API export request**.
2. Let us suppose that we have two UOAs (4 and 14) with the following Research Groups that we wish to export:



4 - Psychology, Psychiatry and Neuroscience

Research groups

Code	Description	
A	CBT	Delete
B	Depression	Delete



14 - Geography and Environmental Studies

Research groups

Code	Description	
A	Geology	Delete
B	Rocks	Delete
C	Air quality	Delete

3. Go to the UAT system: **https://testsubmissionsapi.ref.ac.uk/swagger/index.html**.

4. Enter the value of your **API Key** in accordance with **Authorisation**.
5. Under **Submissions**, click `GET /api/submissions` to expand the endpoint that allows you to find the required `submissionIds`.
6. Click **Try it out**.
7. Click **Execute**.
8. For `unitOfAssessmentIds` 4 and 14, make a note of the `submissionIDs`. In this case, these are `0004efa1-c5a3-4a60-b9fb-3278bcdfd0d8` and `dcba6ea3-9578-45b4-b6d5-6376d7969afb`, respectively.
9. Under **ExportJobs**, click `POST /api/exportjobs` to expand the endpoint that allows you to create an import job.
10. Click **Try it out**.
11. Update the *Export Job* model as in the following example:

```
{
  "sourceFilters": [
    {
     "sourceName": "ResearchGroup",
     "sourceParameters": {"submissionId":"0004efa1-c5a3-4a60-b9fb-
3278bcdfd0d8"}
    }
  ],
   "sourceFilters": [
    {
     "sourceName": "ResearchGroup",
     "sourceParameters": {"submissionId":"dcba6ea3-9578-45b4-b6d5-
6376d7969afb"}
    }
   ],
    "fileFormat": "ExcelXlsx",
    "outputFileName": "UOAs_4_14_Research_Group"
}
```

For the parameters, refer to **API export parameters**.

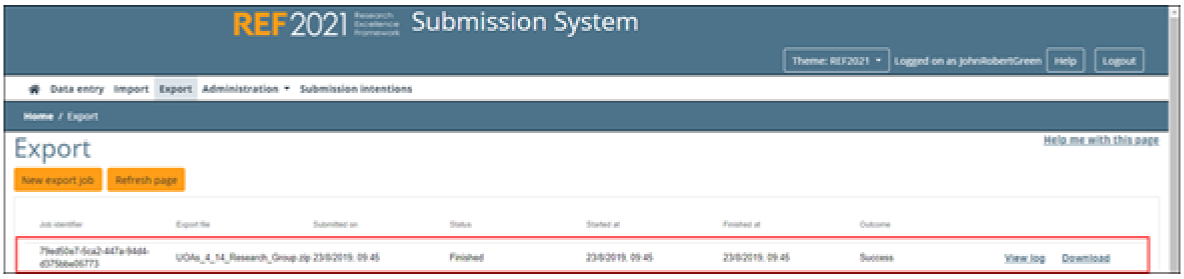
12. Select the required **Response content type** from the drop-down, for example `application/json`.
13. Click **Execute** to submit the request and show the *cURL* that was submitted.
14. Verify the content returned in the **Response body**, and ensure that `isSuccessful` is set to `true`.

 For example:

```
{
   "value": {
       "sources": {},
       "sourceFilters": [
         {
           "sourceName": "ResearchGroup",
           "sourceParameters": {
               "submissionId": "0004efa1-c5a3-4a60-b9fb-3278bcdfd0d8"
         }
       },
         {
           "sourceName": "ResearchGroup",
           "sourceParameters": {
               "submissionId": "dcba6ea3-9578-45b4-b6d5-6376d7969afb"
```

```
          }
        }
    ],
    "fileFormat": "ExcelXlsx",
    "outputFilenameInStore": "UOAs_4_14_Research_Group-79ed50e7-5ca2-447a-94d4-
  d375bbe06773",
    "auditLog": null,
    "log": [],
    "jobId": "79ed50e7-5ca2-447a-94d4-d375bbe06773",
    "tags": "export;forms",
    "status": "Waiting",
    "outcome": "",
    "submittedBy": "JohnRobertGreen",
    "submittedByUserId": "9fccb63f-5c67-4a19-bb33-adcf6471b34f",
    "submittedOn": "2019-08-23T08:45:46.137Z",
    "startedAt": null,
    "finishedAt": null,
    "outputFilename": "UOAs_4_14_Research_Group",
    "lastUpdatedByUserId": "9fccb63f-5c67-4a19-bb33-adcf6471b34f",
    "lastUpdatedBy": "JohnRobertGreen",
    "lastUpdatedOn": "2019-08-23T08:45:45.963Z",
    "versionNumber": 1,
    "processingContext": null
    },
    "isSuccessful": true,
    "isFatal": false,
    "errors": []
  }
```

15. You can now verify the results in the GUI by looking at the export jobs for your user:



Export

Job identifier	Export file	Submitted on	Status	Started at	Finished at	Outcome		
79ed50e7-5ca2-447a-94d4-d375bbe06773	UOAs_4_14_Research_Group.zip	23/8/2019, 09:45	Finished	23/8/2019, 09:45	23/8/2019, 09:45	Success	View log	Download

16. Do either of the following to verify the output.
    - Click **Download** on the far right of the GUI to download the zip file, and then open the Excel file within it. The exported Research Groups are in the **ResearchGroup** worksheet.
    - Use `GET /api/exportjobs/{jobId}/outputfile` and specify the `jobId` given in the **Response body** above.
17. Click **Clear** followed by **Cancel**.
18. Click on the endpoint to collapse it again.

Using the API to create an output with citations

Let us suppose that you wish to use API endpoints to create an output for an **open access** computing article, and attempt a match via the external matching service, in order to retrieve the associated citations.

You can only retrieve citations for an output of type D (Journal article) or E (Conference contribution) that is associated with one of UOAs 1-9, 11 or 16.

Proceed as follows:

1. Go to the UAT system: **https://testsubmissionsapi.ref.ac.uk/swagger/index.html**.
2. Enter the value of your **API Key** in accordance with **Authorisation**.
3. Under **Submissions**, click `GET /api/submissions` to expand the endpoint that allows you to find the `submissionId` under which the output is to be created.
4. Click **Try it out**.
5. Click **Execute**.
6. For `unitOfAssessmentId` 11 (**Computer Science and Informatics**), make a note of the `submissionID`. In this case, it is `d68cdfe0-13a7-4203-8c9b-1877bb8e3705`.
7. Under **Outputs**, click `POST /api/submissions/{submissionId}/outputs` to expand the endpoint that allows you to create an output.
8. Click **Try it out**.
9. Update the *New Output* model accordingly. For example:

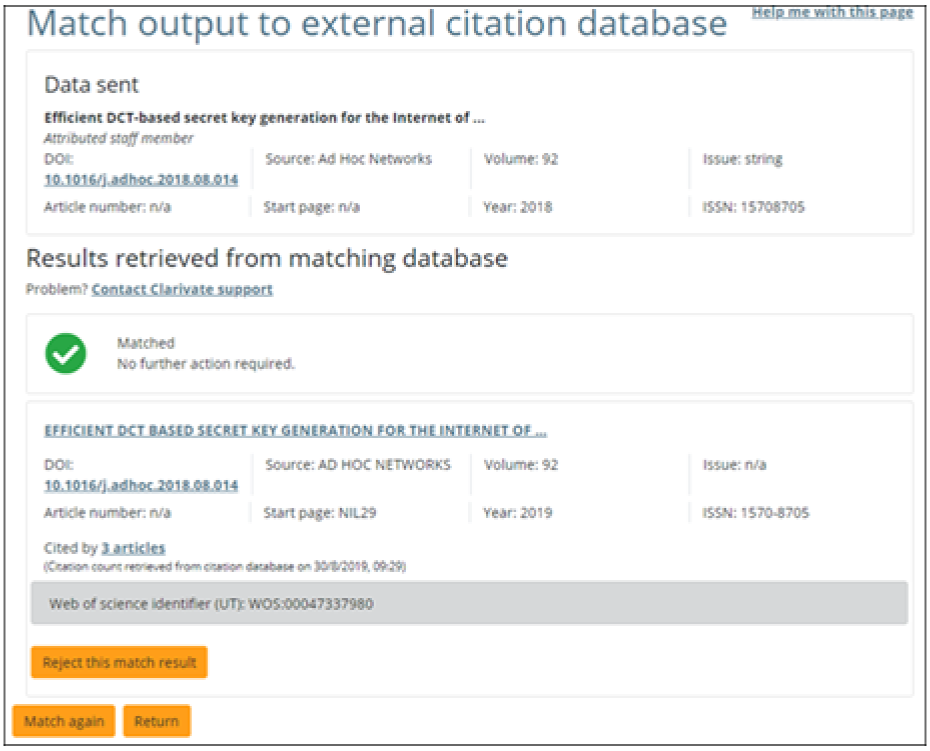
```
{
   "submissionId": "d68cdfe0-13a7-4203-8c9b-1877bb8e3705",
   "outputType": "D",
   "outputIdentifier": "AdHoc_Nets_Vol_92_IOT",
   "title": "Efficient DCT-based secret key generation for the Internet of
   Things",
   "volume": "92",
   "volumeTitle": "Ad Hoc Networks",
   "issue": "string",
   "year": "2018",
   "doi": "10.1016/j.adhoc.2018.08.014",
   "issn": "15708705",
   "openAccessStatus": "Compliant",
   "isPhysicalOutput": false,
   "isPendingPublication": false,
   "isForensicScienceOutput": false,
   "isCriminologyOutput": false,
   "isNonEnglishLanguage": false,
   "isInterdisciplinary": false,
   "proposeDoubleWeighting": false,
   "researchGroup": "A",
   "requiresAuthorContributionStatement": false,
   "isSensitive": false,
   "excludeFromSubmission": false,
   "outputPdfRequired": false
}
```

In order to create an output of type D, you do not need to include all the endpoint parameters listed in the **Swagger UI** documentation, only those listed above. For example, the *auditLog* and *log* properties are not required as the API will add them automatically.

10. Select the required **Response content type** from the drop-down, for example `application/json`.
11. Click **Execute** to submit the request and show the *cURL* that was submitted.
12. Verify the content returned in the **Response body**, and ensure that `isSuccessful` is set to `true`.

Make a note of the `ref2Id`, as you will need this in order to return the citations (in this case, it is `b9badaf0-e59c-415f-8ee2-53eba54748e1`). The user and update information is modified automatically in accordance with the user executing the API.

13. You can now verify the results in the GUI by looking at the outputs for your HEI.
14. Click **Clear** followed by **Cancel**.
15. Click on the endpoint to collapse it again.
16. Under **Citations**, click `POST /api/submissions/{submissionId}/outputs/{ref2Id}/citations/matchrequest` to expand the endpoint that allows you to attempt the citation match for the particular `ref2Id` that has just been created.
17. Click **Try it out**.
18. Enter the `submissionId` where requested (in this case, `d68cdfe0-13a7-4203-8c9b-1877bb8e3705`).
19. Enter the `ref2Id` where requested (in this case, `b9badaf0-e59c-415f-8ee2-53eba54748e1`).
20. Click **Execute**.
21. In the response, look out for a `status` of `SingleMatch`, with the `matches` block returning the associated `webOfScienceIdentifier` and number of citations.
22. Click **Clear** followed by **Cancel**.
23. Click on the endpoint to collapse it again.
24. You can now verify the results in the GUI by looking at the *Match output to external citation database* page:



Match output to external citation database

Data sent

Efficient DCT-based secret key generation for the Internet of ...

Attributed staff member

DOI: 10.1016/j.adhoc.2018.08.014	Source: Ad Hoc Networks	Volume: 92	Issue: string
Article number: n/a	Start page: n/a	Year: 2018	ISSN: 15708705

Results retrieved from matching database

Problem? Contact Clarivate support

Matched
No further action required.

EFFICIENT DCT BASED SECRET KEY GENERATION FOR THE INTERNET OF ...

DOI: 10.1016/j.adhoc.2018.08.014	Source: AD HOC NETWORKS	Volume: 92	Issue: n/a
Article number: n/a	Start page: NIL29	Year: 2019	ISSN: 1570-8705

Cited by 3 articles
(Citation count retrieved from citation database on 30/8/2019, 09:29)

Web of science identifier (UT): WOS:00047337980

25. Alternatively, you can use `GET /api/submissions/{submissionId}/outputs/{ref2Id}/citations/matchrequest` under **Citations** to retrieve the details of the match request, as follows:
    a. Expand the endpoint.
    b. Click **Try it out**.
    c. Enter the `submissionId` where requested (in this case, `d68cdfe0-13a7-4203-8c9b-1877bb8e3705`).
    d. Enter the `ref2Id` where requested (in this case, `b9badaf0-e59c-415f-8ee2-53eba54748e1`).
    e. Click **Execute**.
    f. The format of the response is given in the following extract:

```
{
   "ref2Id": "b9badaf0-e59c-415f-8ee2-53eba54748e1",
   "submissionId": "d68cdfe0-13a7-4203-8c9b-1877bb8e3705",
   "outputType": "D",
   "status": "SingleMatch",
   "statusComments": null,
   "webOfScienceIdentifier": "WOS:00047337980",
   "surname": null,
   "title": "Efficient DCT-based secret key generation for the Internet
   of Things",
   "volumeTitle": "Ad Hoc Networks",
   "volume": "92",
   "issue": "string",
   "firstPage": null,
   "articleNumber": null,
   "issn": "15708705",
   "doi": "10.1016/j.adhoc.2018.08.014",
    "year": "2018",
    "matches": [
        {
          "webOfScienceIdentifier": "WOS:00047337980",
          "probability": 0.977903068,
          "title": "EFFICIENT DCT BASED SECRET KEY GENERATION FOR THE
          INTERNET OF THINGS",
          "sourceTitle": "AD HOC NETWORKS",
          "volume": "92",
          "issue": null,
          "startPage": "NIL29",
          "endPage": null,
          "numberOfPages": 11,
          "articleNumber": null,
          "issn": "1570-8705",
          "doi": "10.1016/j.adhoc.2018.08.014",
          "year": "2019",
          "timesCited": 3,
...
...
...
}
```

g. Click **Clear** followed by **Cancel**.

h. Click on the endpoint to collapse it again.