Citation Generation API Contract

Scope & UI: **Citation generator**

Usecase:

To generate citations based on the below 4 options:

- Import from Zotero
- Free Search
- Add a new item to Zotero Library based on the free search
- Enter data manually - TBD
- Import From file - *will be picked up in the next phase*

API Templates:

Option 1: Import from Zotero

- Here the user will be redirected to the zotero login page
- After successful login, user will be asked to create a new API key which can have read/write access to the users library / collections
- An auth token will be generated to authenticate the user with the help of OAuth v1 flow.

API endpoint to generate auth token : BASE URL + ‘ /zotauth’

Method: ‘POST’

Request Body:

```
{
  "name": "zotero",
  "consumer_key": "a3XXXXXXXXXX",
  "consumer_secret": "1cXXXXXXXXXX",
  "request_token_url": "https://www.zotero.org/oauth/request",
  "access_token_url": "https://www.zotero.org/oauth/access",
  "authorize_url": "https://www.zotero.org/oauth/authorize",
  "base_url": "https://api.zotero.org"
}

```

Response Body:

```
{
  "request_token": "<request_token>",
  "request_token_secret": "<request_token_secret>",
  "authorize_url": "<authorize_url>"
}
```

API endpoint to authenticate user : BASE URL + ‘ /verify’

Method: ‘POST’

Request Body: To authenticate the user

```
{
  "token": "<token>",
  "request_token": "<request_token>",
  "request_token_secret": "<request_token_secret>"
}
```

Response Body: Success

```
[
  {
    'key': 'TESF7ITX',
    'version': 106,
    'library': {
      'type': 'user',
      'id': 11222345,
      'name': 'govindrs',
      'links': {
        'alternate': {
          'href': 'https://www.zotero.org/govindrs',
          'type': 'text/html'
        }
      }
    },
    'links': {
      'self': {
        'href': 'https://api.zotero.org/users/11222345/items/TESF7ITX',
        'type': 'application/json'
      },
      'alternate': {
        'href': 'https://www.zotero.org/govindrs/items/TESF7ITX',
        'type': 'text/html'
      }
    },
    'meta': {
      'numChildren': 0
    },
    'data': {
      'key': 'TESF7ITX',
      'version': 106,
      'itemType': 'webpage',
      'title': 'Csl search by example',
      'creators': [

      ],
      'abstractNote': '',
      'websiteTitle': '',
      'websiteType': '',
      'date': '',
      'shortTitle': '',
      'url': 'https://editor.citationstyles.org/searchByExample/',
      'accessDate': '2023-03-29T06:54:41Z',
      'language': '',
      'rights': '',
      'extra': '',
      'tags': [

```

```
      ],
      'collections': [
        'E63G4WCI'
      ],
      'relations': {

      },
      'dateAdded': '2023-03-29T06:56:41Z',
      'dateModified': '2023-03-29T06:56:41Z'
    }
  },
]

```

Response Body: Failure

```
{
  "message": "Error in Zotero OAuth token exchange"
}

```

Option 2: Free Search

- Here the user can generate citations by entering either a DOI, journal/book title or PMID (pubmed_id) in the search bar
- Based on the input we will search the data in our Dyson DB using Elastic Search query
- The user will need to specify a citation style here which is a mandatory parameter

API endpoint : BASE URL + ‘ /search’

Method: ‘POST'

Headers: *mandatory*

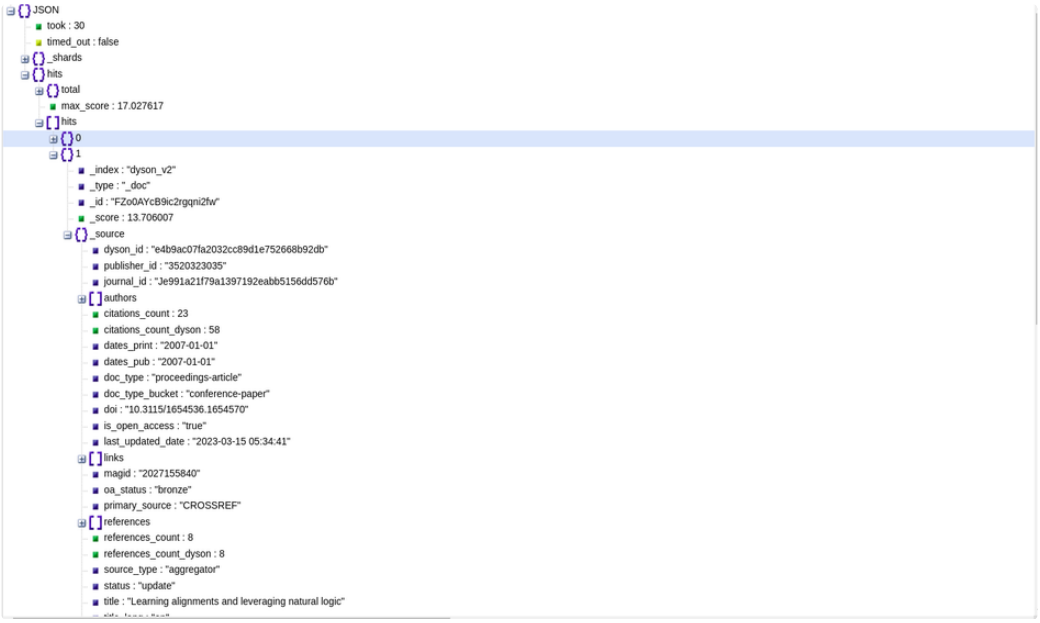
```
{
  Content-Type: application/json
}
```

Request Body:

```
{
  "index": "dyson_v2",
  "payload": {
    "query": {
      "bool": {
        "must": [
          {
            "match": {
              "doi": "<doi>"
            }
          }
        ]
      }
    }
  }
}

```

Response Body: Since dyson success response is too big, have just added a screenshot



Option 3: Based on the free search a user can add a new item to its's Zotero library if required

API Endpoint: BASE URL + '/zotero/items/create'

Method: POST

Headers: *mandatory*

```
{
  Authorization: <Zotero api_key generated in step 1>
  Content-Type: application/json
}
```

Request Body:

```
{
  "items": [
    {
      "itemType": "<itemType>",
      "title": "<title>",
      "creators": [
        {
          "firstName": "<firstName>",
          "lastName": "<lastName>",
          "creatorType": "<creatorType>"
        }
      ],
      "publicationTitle": "<publicationTitle>",
      "date": "<date>",
      "volume": "<volume>",
      "pages": "<pages>",
      "DOI": "<doi>"
    }
  ]
}
```

```
21
```

Response Body:

```
{
  "success": true,
  "successful": [
    {
      "key": "<item_key>"
    }
  ],
  "failed": []
}
```

Option 4: Enter data manually

- Here the user wil enter the data in the form provided. Based on the different article types the form fields will vary.

API Endpoint: BASE URL + '/<article_type>_form'

Method: POST

Headers: *mandatory*

```
{
  Authorization: <Zotero api_key generated in step 1>
  Content-Type: application/json
}
```

Request Body: based on the article type the request body will have varying keys

```
{
  "fetch_id": "",
  "article_type": "journal/book/website"
  "citation_style": "",
  "title": "",
  "author": "",
  ...
  ...
  ...
}
```

Response body: Similar to the common response body defined below

Common Request Body - *TBD*

API endpoint: BASE URL + ' /citation_generator'

Headers: *mandatory*

```
{
  "Content-Type": "application/json",
  "Authorization":  "<static_bearer_token>"
}
```

Request Body:

```
{
    "fetch_id": "",
    "user_type": "zotero/non-zotero",       ##  mandatory
    "user_data": {       ## this will be empty in case of non zotero users
        "id": "11222345",
        "name": "govindrs",
        "item_key": "9UXLMG8V"
        "api_key": "<api_key>""
    },
    "citation_style": "APA",   ## this is a mandatory key
    "doi/title/pubmedis": "doi/title/35007113",    # pass any one of these
}
```

Citations styles

- Based on the citation style given by the user we will be using citproc python package to generate citations for the data received from zotero and dyson.
- We need to design the input in the below format for citeproc to generate citations for any given data/input

```
citeproc_input_json = '''
[
    {
        "author": [
            {
                "family": "Schmidhuber",
                "given": "Jürgen"
            }
        ],
        "id": "ITEM-4",
        "issued": {
            "date-parts": [
                [2015]
            ]
        },
        "title": "Deep learning in neural networks: An overview",
        "container-title": "Neural Networks",
        "volume": "61",
        "page": "85-117",
        "type": "article-journal",
        "DOI": "10.1016/j.neunet.2014.09.003",
        "URL": "https://doi.org/10.1016%2Fj.neunet.2014.09.003",
        "publisher": "Elsevier {BV}"
    },
    {
        "id": "ITEM-5",
        "type": "book",
        "call-number": "SB419.5 .D85 2008",
        "edition": "Rev. and updated ed., [2nd ed.]",
        "event-place": "Portland, Or",
        "ISBN": "978-0-88192-911-9",
        "number-of-pages": "328",
        "publisher": "Timber Press",
        "publisher-place": "Portland, Or",
        "source": "Library of Congress ISBN",
        "title": "Planting green roofs and living walls",
        "author": [
            {
```

```
                "family": "Dunnett",
                "given": "Nigel"
            },
            {
                "family": "Kingsbury",
                "given": "Noël"
            }
        ],
        "issued": {
            "date-parts": [
                [
                    "2008"
                ]
            ]
    }
]
'''
```

Common Response Body

- Response Status 200

```
{
    "status": "success",
    "fetch_id": "",
    "request_data": {
        "user_type": "zotero/non-zotero",       ##  mandatory
        "user_data": {       ## this will be empty in case of non zotero users
            "id": "11222345",
            "name": "govindrs",
            "item_key": "9UXLMG8V"
            "api_key": "<api_key>""
        },
        "citation_style": "APA",   ## this is a mandatory key
        "doi/title/pubmedis": "doi/title/35007113",    # pass any one of these
    },
    "response_data": {
        "items": [{
            "item_id": "1",
            "item_data": {},
            "citations": "",
            "biblography": "",
            "scite_badge": ""
        },
        {
            "item_id": "2",
            "item_data": {},
            "citations": "",
            "biblography": "",
            "scite_badge": ""
        }
        ]
    }
}

```

- Response Status 400

```
{
  "status": "failed",
  "message": "Invalid request id provided."
}
```

- Response Status 401

```
{
  "status": "failed",
  "message": "Authorization failed."
}
```

- Response Status 403

```
{
  "status": "failed",
  "message": "Missing Authorization token."
}
```

Scite Badge


```
<div class="scite-badge"
     data-doi="10.1016/j.biopsych.2005.08.012"
     data-layout="horizontal"
     data-show-zero="false"
     data-small="false"
     data-show-labels="false"
     data-tally-show="true">
</div>
<script async type="application/javascript" src="https://cdn.scite.ai/badge/scite-badge-latest.min.js">
</script>
```

TO DO:

- [ ] Create compound citations
- [ ] Edit citations dynamically