Figshare API Workshop

Open Repositories 2023
Stellenbosch, South Africa
Twitter: #openrepos2023

Quick Activity

Photo by iorni.com on Unsplash

Instructors

Adrian-Tudor Pănescu
Technical Team Leader

Andrew Mckenna-Foster
Product Specialist (Assistant instructor)

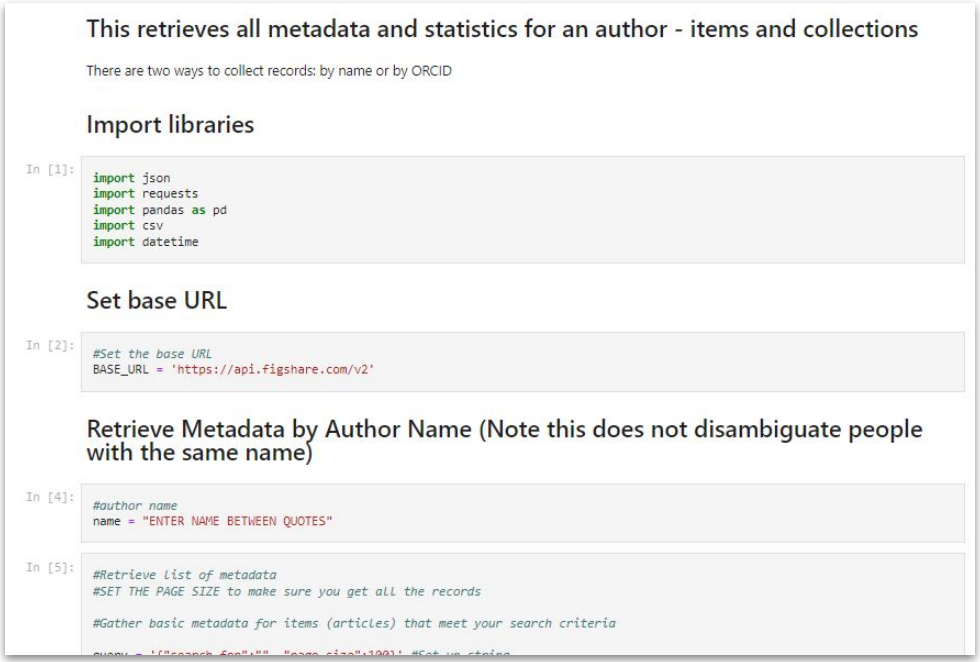
Photo by Neil Thomas on Unsplash

This workshop is an opportunity to create something useful to you

Examples

A personal statistics report

Try this in Google Colab.



This retrieves all metadata and statistics for an author - items and collections

There are two ways to collect records: by name or by ORCID

Import libraries

In [1]:

```
import json
import requests
import pandas as pd
import csv
import datetime
```

Set base URL

In [2]:

```
#Set the base URL
BASE_URL = 'https://api.figshare.com/v2'
```

Retrieve Metadata by Author Name (Note this does not disambiguate people with the same name)

In [4]:

```
#author name
name = "ENTER NAME BETWEEN QUOTES"
```

In [5]:

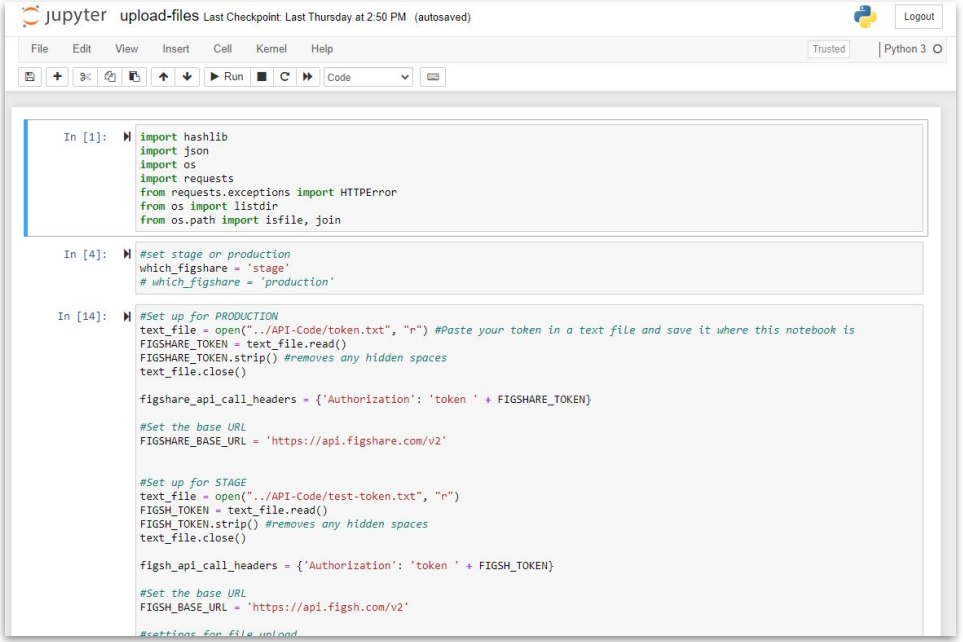
```
#Retrieve list of metadata
#SET THE PAGE SIZE to make sure you get all the records

#Gather basic metadata for items (articles) that meet your search criteria
```

Examples

Manage repository records

Wesleyan University uses the API to manage records and upload large file collections

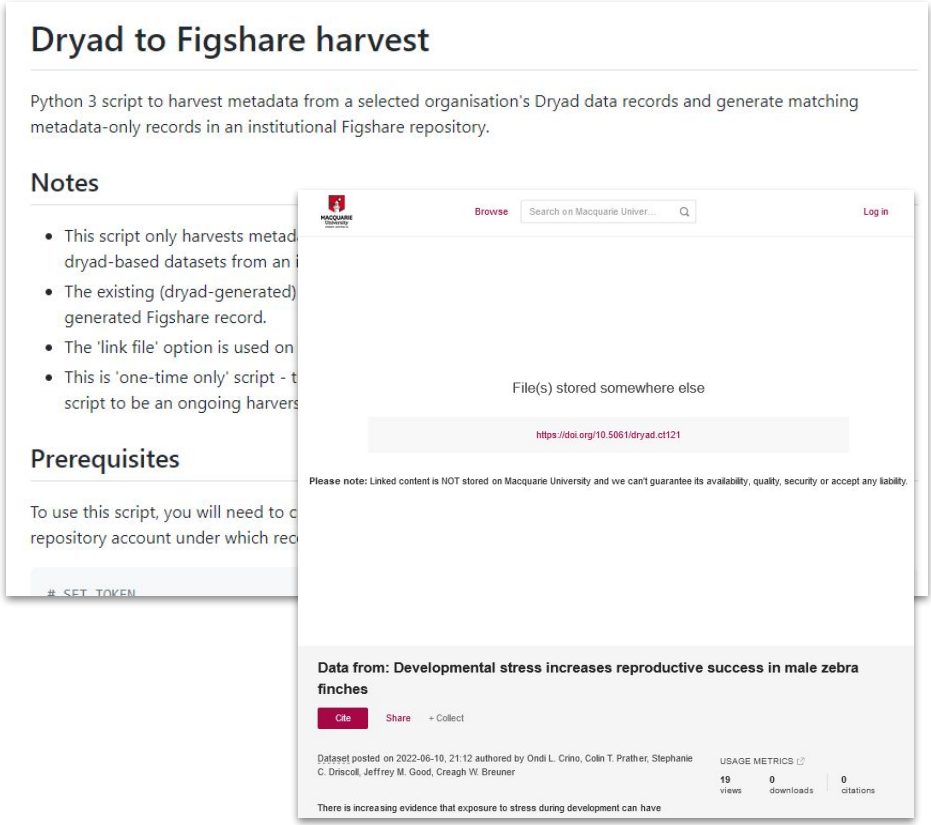


Examples

Harvest records to create a data catalog

Either from another repository to Figshare or from Figshare to another repository

Macquarie University built a tool to harvest metadata from Dryad and deposit in their repository as linked file records

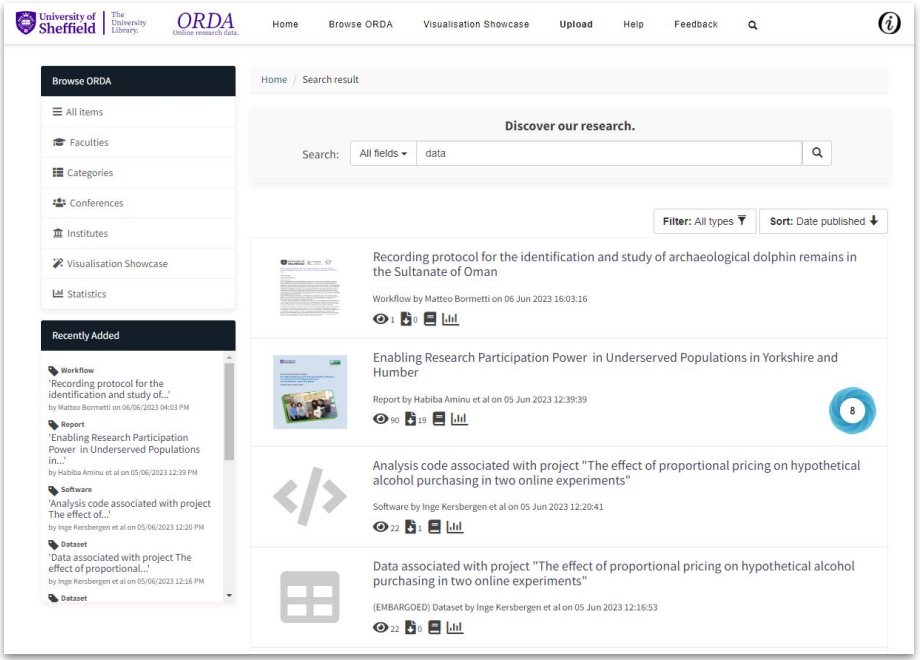


https://github.com/mq-eresearch/dryad_to_figshare/tree/v1.0.0

Examples

Build a custom web application

University of Sheffield built a custom search interface: https://orda.shef.ac.uk/



Please note down 1-2 projects you'd like to work on

As we progress through the workshop apply what you learn to your project

You may not finish the project today, but should leave with a clear roadmap

Photo by No Revisions on Unsplash

Workshop Website

https://amckennafoster.github.io/figshare-api-workshop.github.io/index.html

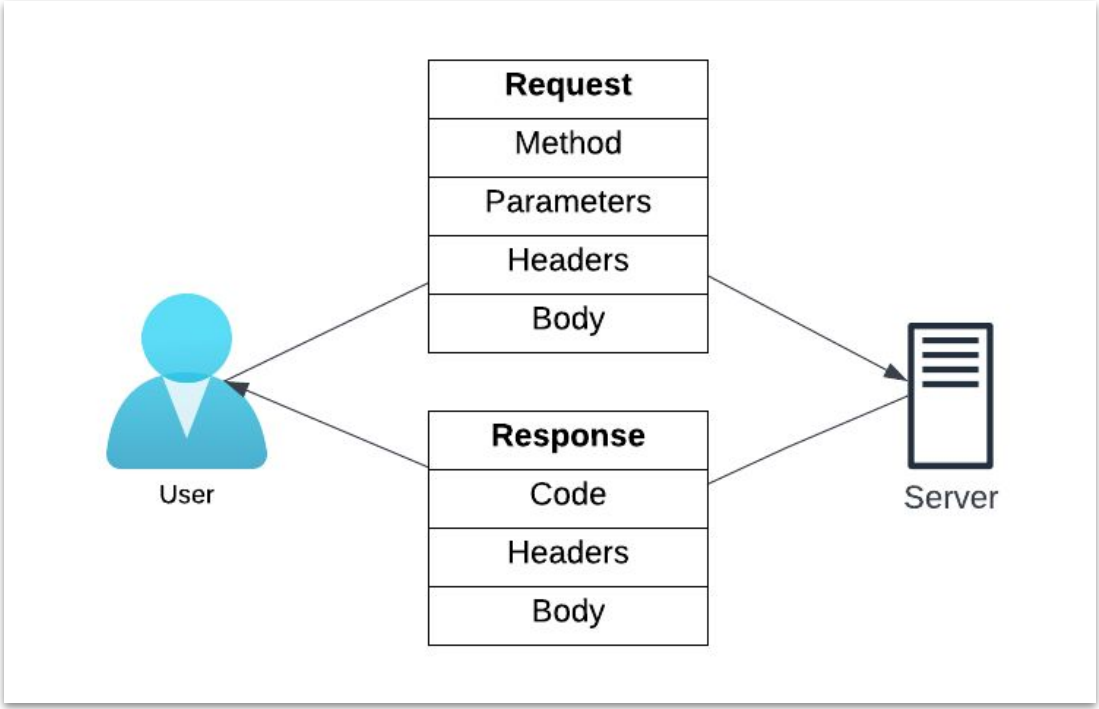
- Schedule
- Links
- Resources

Workshop Outline

- Introduction to HTTP requests and APIs
- Introduction to Figshare's API(s)
- Accessing Figshare's REST API using Postman
- Building an application over Figshare's REST API

HTTP Requests and APIs

What is a HTTP request?



What is a HTTP request?

```
POST /v2/articles/search?offset=10&limit=1 HTTP/1.1

Host: api.figshare.com

Content-Type: application/json

Authorization: Bearer b2be49036a3158c5edd5a0553ae9


{"search_for": ":tags: test"}
```

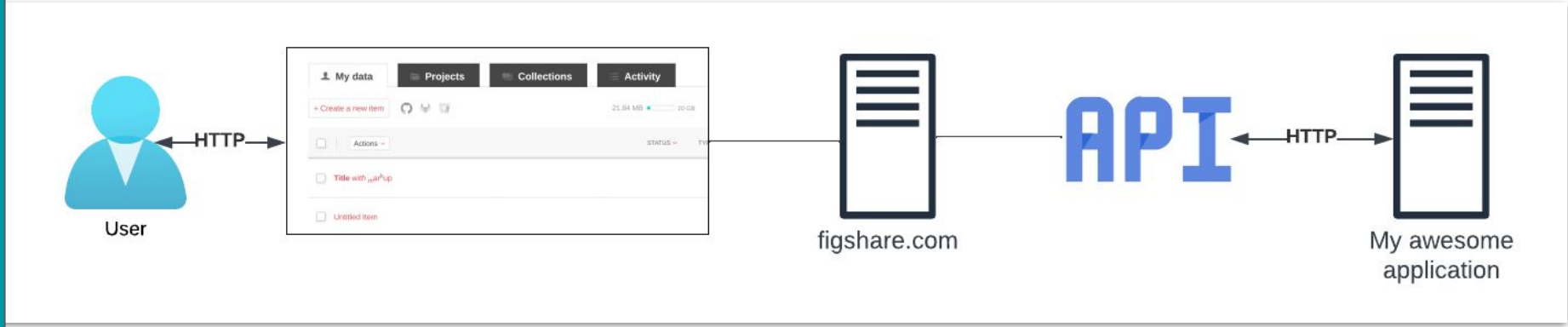
HTTP is stateless

- Requests are independent from one another
- Each request needs to contain enough information in order for the server to be able to respond to it
- This is why, for example, you need to send authentication information with each HTTP request

What is an API?

- Abbreviation for **Application Programming Interface**
- It’s an interface which allows other applications to use the functionality of a software system

What is an API?



What is REST?

- Abbreviation for **REpresentational State Transfer**
- A set of principles on how web APIs should be built
- Servers should always respond with the representation of a *resource*

Example of a REST resource


```
{
    "files": [
        {
            "id": 25133441,
            "name": "Panescu_Advances_in_Digital_Repositories_PhD_Thesis.pdf",
            "size": 1411929,
            "is_link_only": false,
            "download_url": "https://ndownloader.figshare.com/files/25133441",
            "supplied_md5": "87f39b4da1a371572523b59bf83bb32e",
            "computed_md5": "87f39b4da1a371572523b59bf83bb32e"
        }
    ],
    "custom_fields": [],
    "authors": [
        {
            "id": 1402906,
            "full_name": "Adrian-Tudor Panescu",
            "is_active": true,
            "url_name": "Adrian-Tudor_Panescu",
            "orcid_id": "0000-0002-8940-898X"
        }
    ],
    "figshare_url": "https://figshare.com/articles/thesis/Advances_in_Digital_Repositories/8247626",
    "description": "This thesis presents the evolution of research repositories, from collections of peer-reviewed outputs, such as
        journal articles or monographs, to collections holding a wide array of materials, including data sets, preprints, scientific
        software or protocols, based on the new realities of the scientific research endeavour. It also discusses a number of novel
        practical solutions to repository issues such as licencing, bibliographic record modelling, dissemination and linking, or
        record management and migration, by employing new technologies such as linked data, blockchain, and extract, transform and
        load frameworks.",
    "funding": null,
    "funding_list": [],
    "version": 2,
    "status": "public",
```

What is JSON?

- Abbreviation for **JavaScript Object Notation**
- Human-readable data interchange format
- Consists of key-value pairs, lists and other basic types (numbers, strings)
- For example: `{"name": "Jane Doe", "someList": [1, "two", 3]}`

REST and request methods (verbs)

- GET: retrieve a resource
- POST: add a new resource
- PUT: update a new resource
- PATCH: partially update a new resource
- DELETE: remove a resource

HTTP status codes

- Standard (and easiest) way of understanding what happened with your requests
- Some common status codes:
  - 200 OK
  - 400 Bad request
  - 403 Forbidden
  - 404 Not Found
  - 500 Internal Server Error
- Full list: https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

HTTP request body

- The actual data sent to the server
- It’s any sequence of bytes (JSON, XML, an image file) but…
- We need to tell the server what we are sending:
  - `Content-Type: application/json`
- The server response can also contain any sequence of bytes but…
- We can tell the server what we accept:
  - `Accept: */*`
- The request/response bodies are optional

HTTPS

- Secure HTTP
- A secure channel is created between the client and the server
- Request details (URL, headers, body) are not visible on the open network
- Important if sensitive data (e.g., authentication credentials) is exchanged

Tea Break
30 min

Photo by TeaCora Rooibos on Unsplash

The Figshare API

Figshare API(s)

- Figshare actually has 4 different *APIs*:
  - REST API: https://api.figshare.com/v2
  - Stats API: https://stats.figshare.com/
  - OAI-PMH: https://api.figshare.com/v2/oai
  - ResourceSync: https://scholardata.sun.ac.za/.well-known/resourcesync

Figshare API(s)

- REST API:
  - v1: not very RESTful, deprecated in 2021
  - v2: current version, centered around resources, backwards-compatible
  - v3: will better expose all UI functionality, improve documentation, better integrate stats
- Stats API:
  - Distinct due to the fact that stats are held in a different database and processed by different services than metadata
- OAI-PMH:
  - implemented in order to allow harvesting all Figshare items
- ResourceSync:
  - Another standard from OAI allowing synchronization between systems
  - Used for implementing sitemaps

ACTIVITY

Try GET request

Instructions to retrieve full metadata and retrieve views are here: https://amckennafoster.github.io/figshare-api-workshop.github.io/workshop/workshop-api-basics.html

OpenAPI and Swagger

- OpenAPI
  - A specification language for REST APIs
  - Allows prototyping and generating documentation, client code and test cases
- Swagger:
  - OpenAPI implementation
  - Editor: https://editor.swagger.io/
  - Swagger UI: https://swagger.io/tools/swagger-ui/

Figshare and OpenAPI

- Documentation publicly available at https://docs.figshare.com
- Source code for documentation: https://github.com/figshare/user_documentation
- Swagger file: https://github.com/figshare/user_documentation/blob/master/swagger_documentation/swagger.json

API authentication

- REST API - personal token: https://help.figshare.com/article/how-to-get-a-personal-token
    - OAuth2 - create applications which require users to log into their own Figshare account
- Stats API - user/password, contact support@figshare.com for an account
- Credentials are sent to the API server via the `Authorization` header
- **Make sure credentials are stored in a secure location!**

ACTIVITY

Authenticate and try a POST request

https://amckennafoster.github.io/figshare-api-workshop.github.io/workshop/workshop-api-basics.html

Sandbox accounts

global.user@figsh.com.bk
Bongani Jwara
Charl Roberts
Nambitha Manqola
Nkululeko Magwaza

global.user2@figsh.com.bk
Andisiwe Magocoba
Bubele Bido
Janina Van der Westhuizen
Yan Han

global.user3@figsh.com.bk
Samuel Simango
Songezo Mpikashe
Sizwe Ngcobo

https://global.figsh.com

global.user4@figsh.com.bk
Regina Sikhosana
Richard Nobebe
Xabiso Xesi

global.user5@figsh.com.bk
Eddie Mathiba
Martin Dreyer
Motlanalo Hlophe
Tshinakaho Malesa

DEMO

A quick overview of the Figshare REST API resources

Live repository: https://docs.figshare.com/
Sandbox repository: https://docs.figsh.com/

Q&A & PROJECT TIME

Use this time to ask questions and lay out the requirements for your project(s):

1. What is your objective
2. What information will you need to retrieve or send
3. What endpoints will you need to use

Photo by Ella Olsson on Unsplash

Performing Figshare API requests

Postman

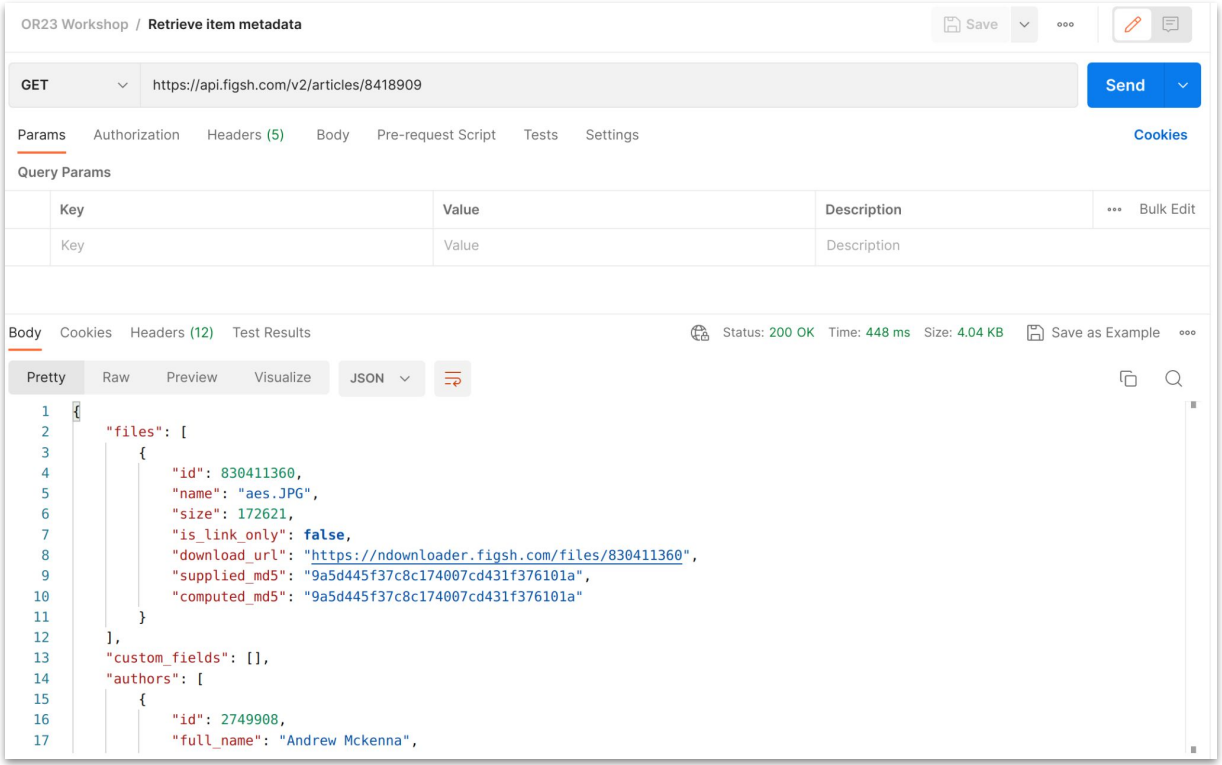
- Tool for performing API requests
- Can be downloaded from https://www.postman.com/downloads/
  - Web version available at https://web.postman.co/
- All demo requests available as a Postman collection: https://www.postman.com/tudor/workspace/public-workspace/collection/329625-1a755e1d-811a-48b8-8471-860e3f09f0ec

DEMO

Using Postman to perform requests

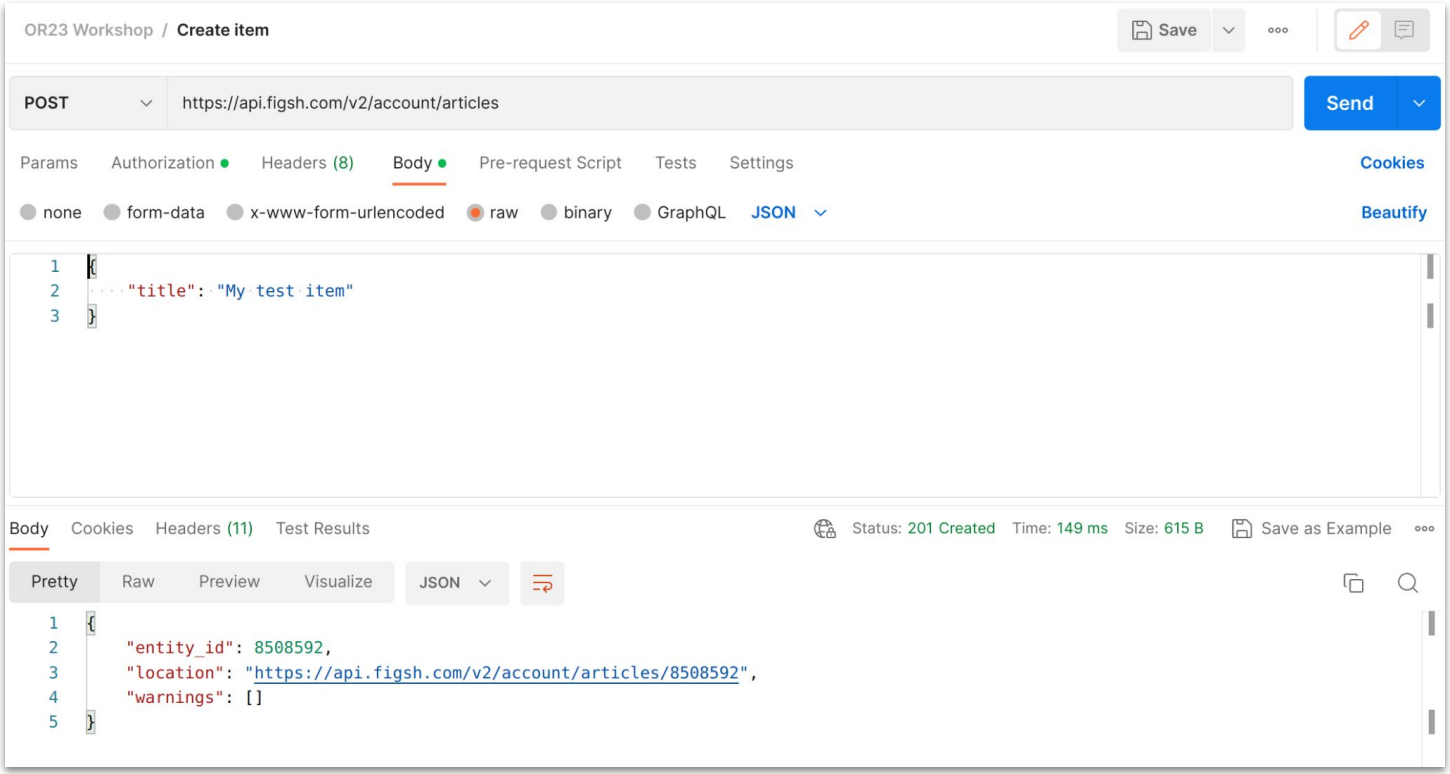
https://amckennafoster.github.io/figshare-api-workshop.github.io/workshop/postman-use-api.html

Retrieve item metadata

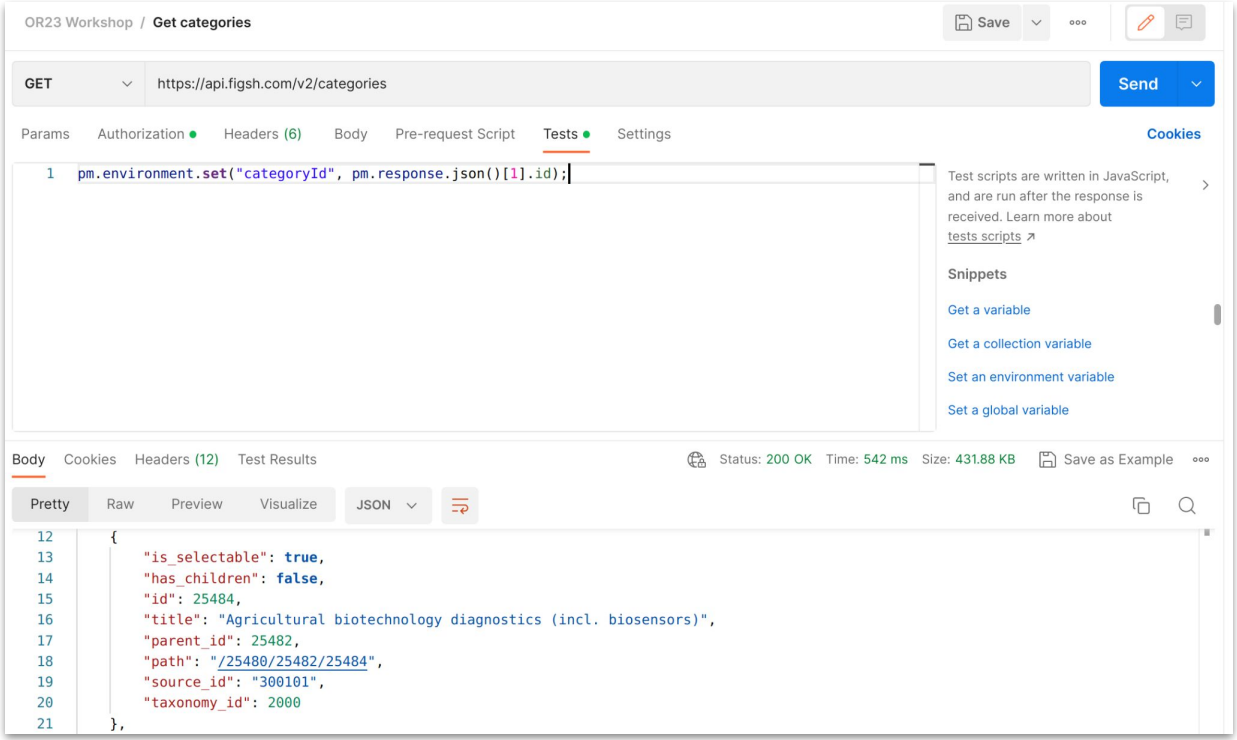


Create > Add metadata > Publish

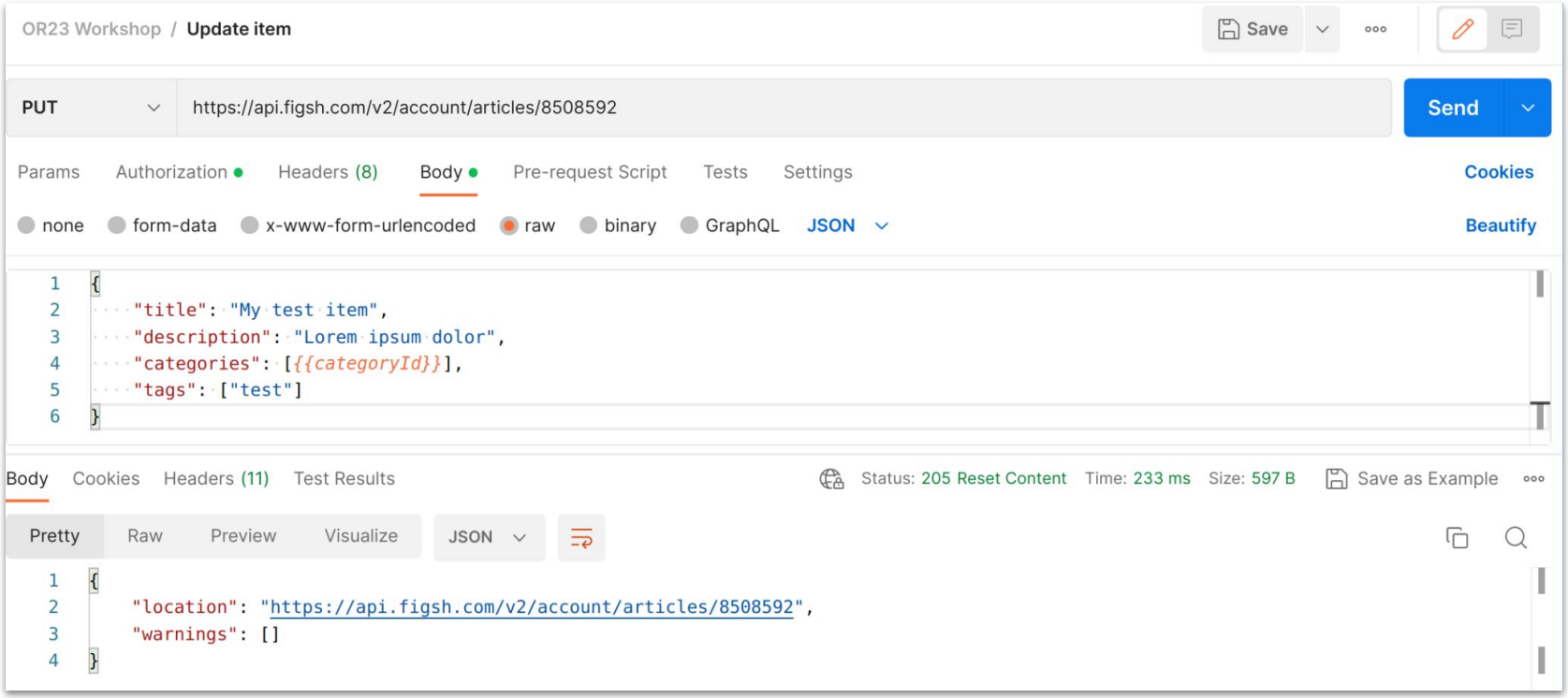
Create item


OR23 Workshop / Create item
Save
POST
https://api.figsh.com/v2/account/articles
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"title": "My test item"
}
Body
Cookies
Headers (11)
Test Results
Status: 201 Created
Time: 149 ms
Size: 615 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
{
"entity_id": 8508592,
"location": "https://api.figsh.com/v2/account/articles/8508592",
"warnings": []
}

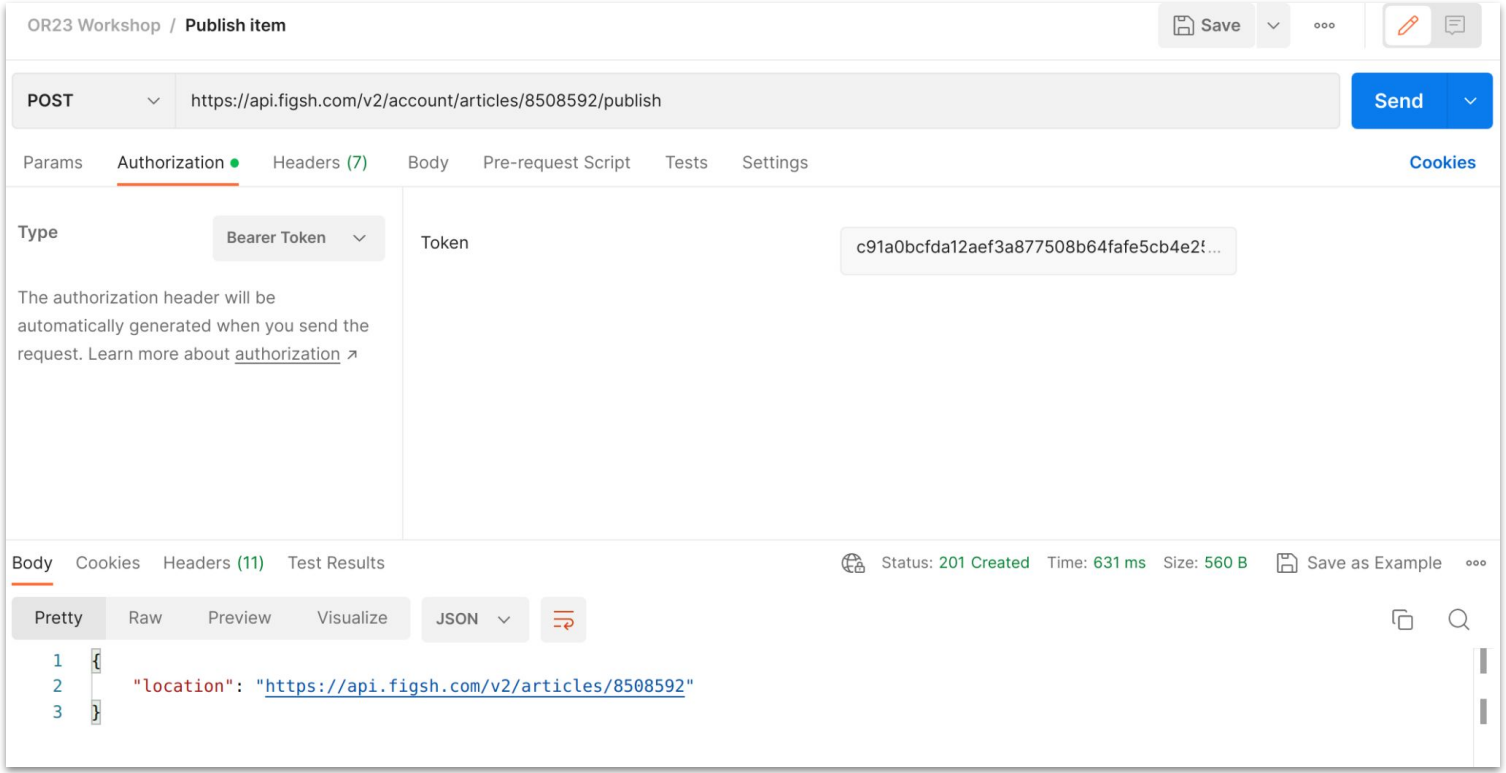
Retrieve categories



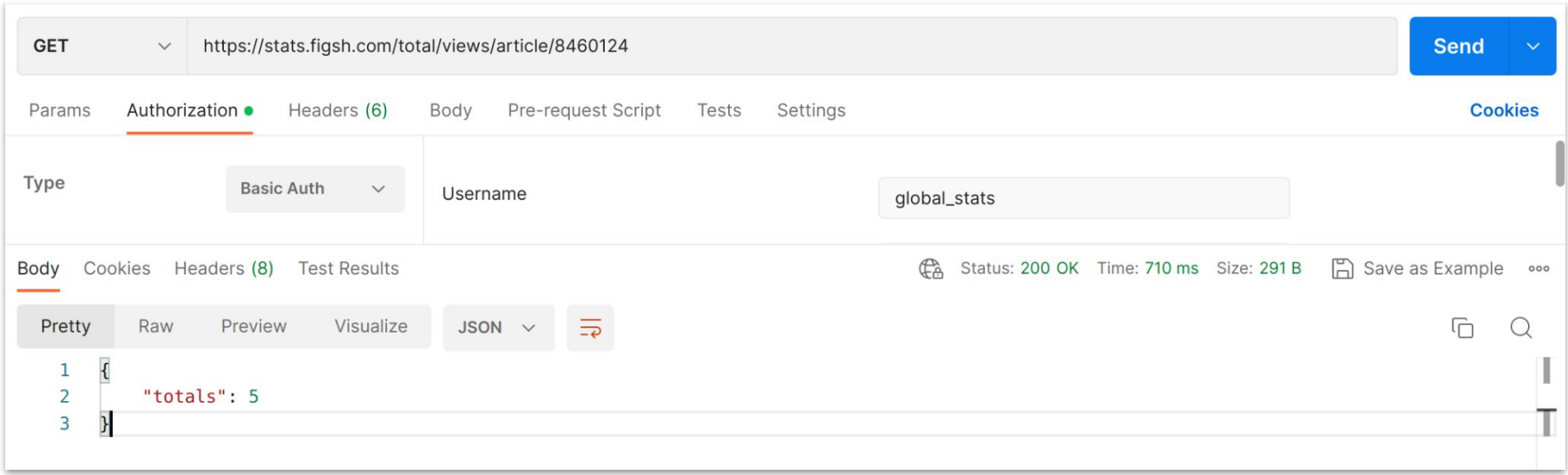
Update item


OR23 Workshop / Update item
Save
PUT
https://api.figsh.com/v2/account/articles/8508592
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"title": "My test item",
"description": "Lorem ipsum dolor",
"categories": [{{categoryId}}],
"tags": ["test"]
}
Body
Cookies
Headers (11)
Test Results
Status: 205 Reset Content
Time: 233 ms
Size: 597 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
{
"location": "https://api.figsh.com/v2/account/articles/8508592",
"warnings": []
}

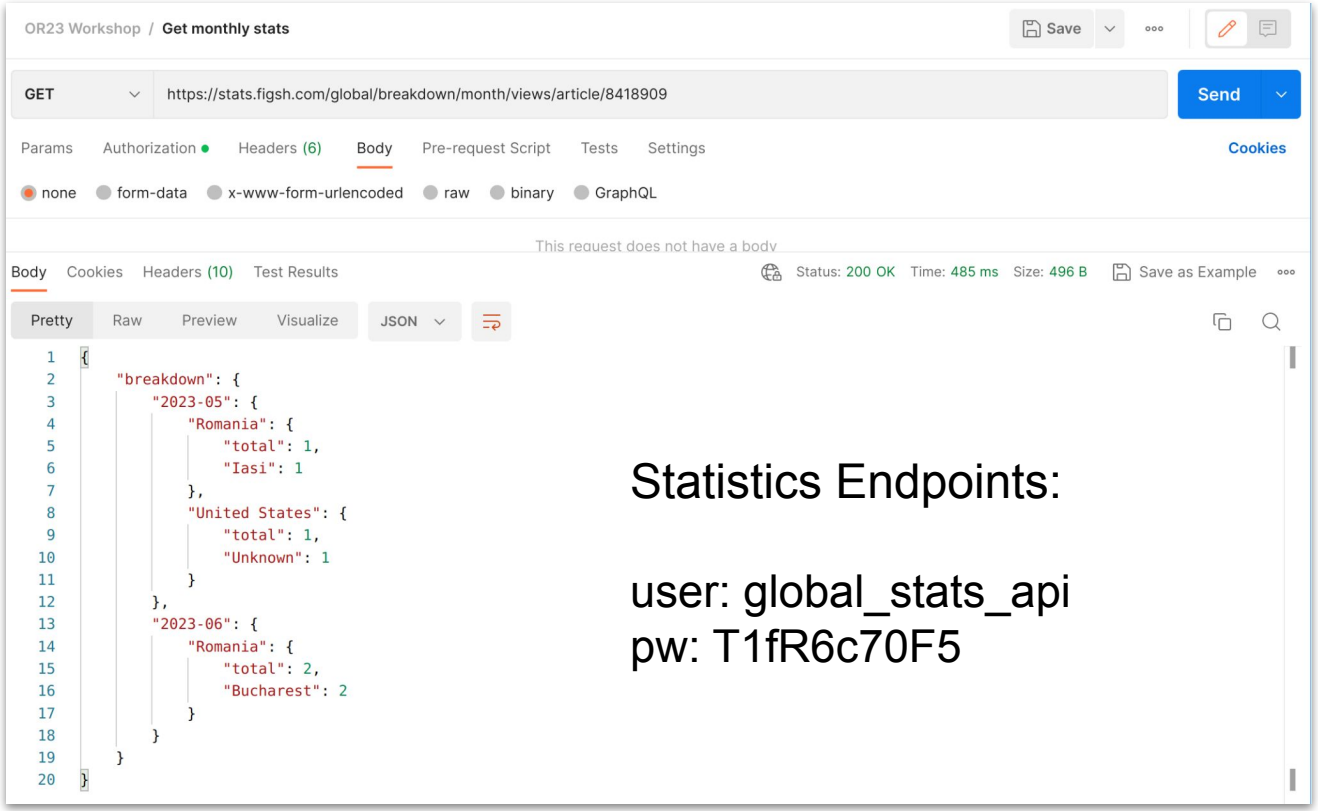
Publish item



Retrieve views



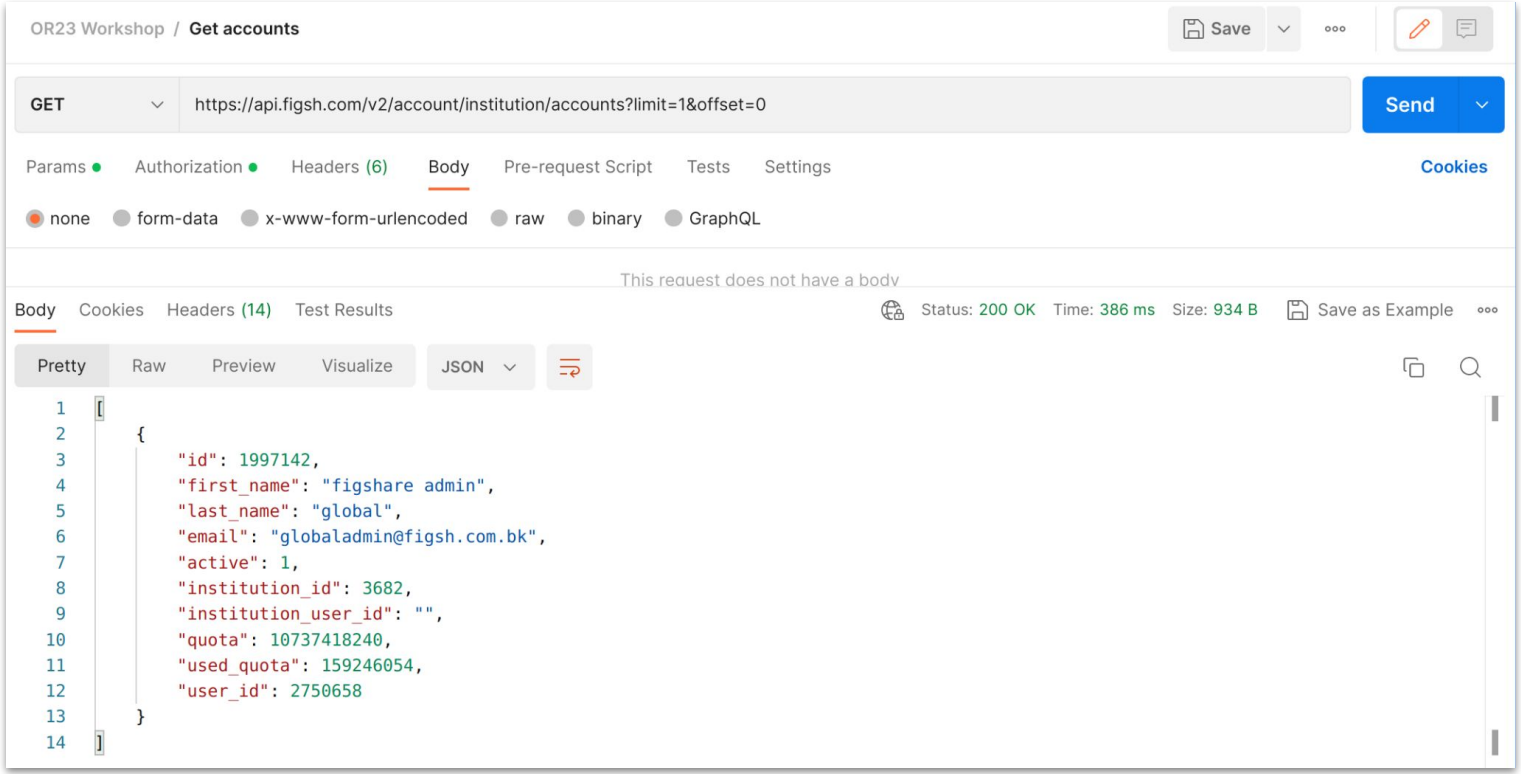
Retrieve monthly views breakdown



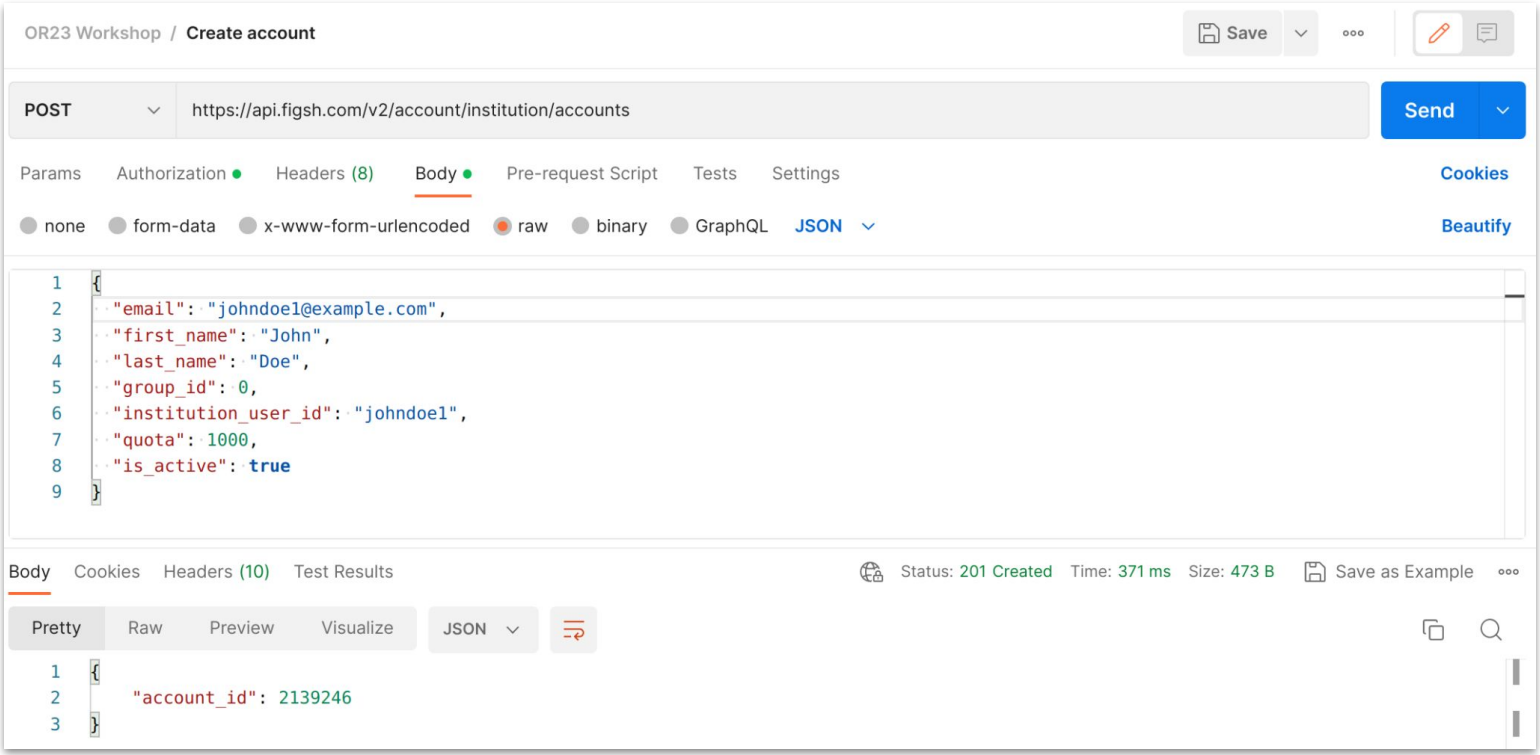
Statistics Endpoints:

user: global_stats_api
pw: T1fR6c70F5

Retrieve repository accounts



Create account



PROJECT TIME

Postman

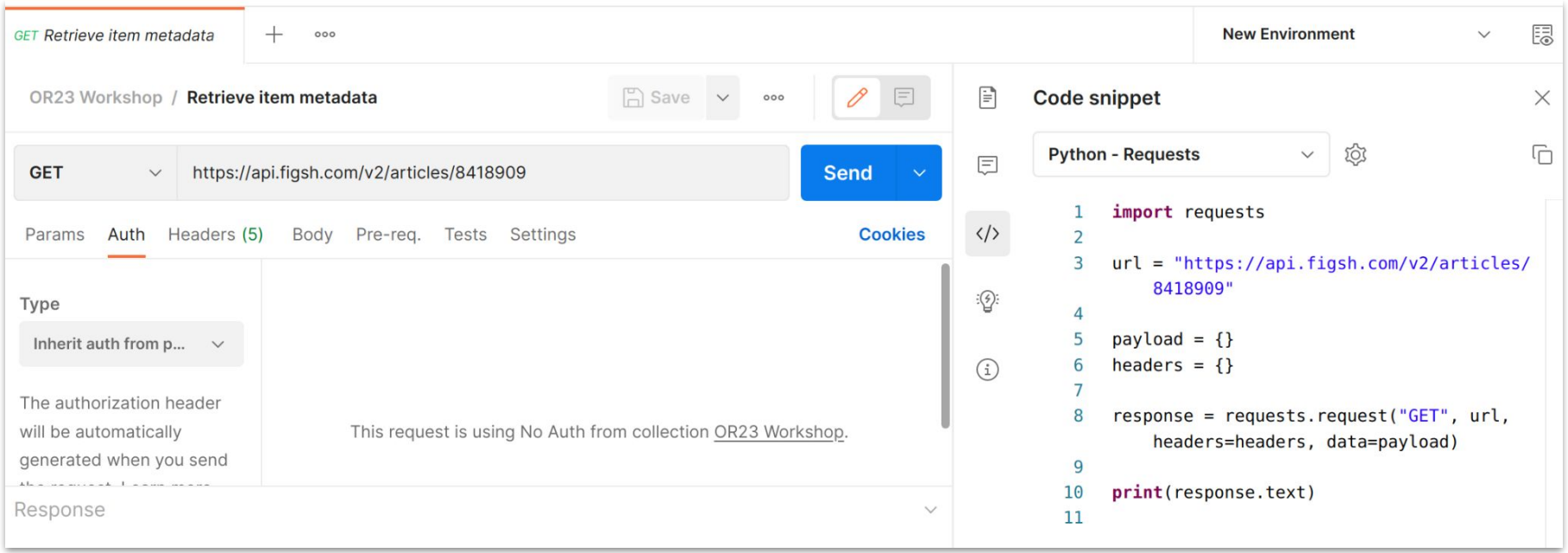
Use Postman to test out the endpoints you need for your project.

If your project requires chaining API calls, decide what language you will use. There are resources for Python here.

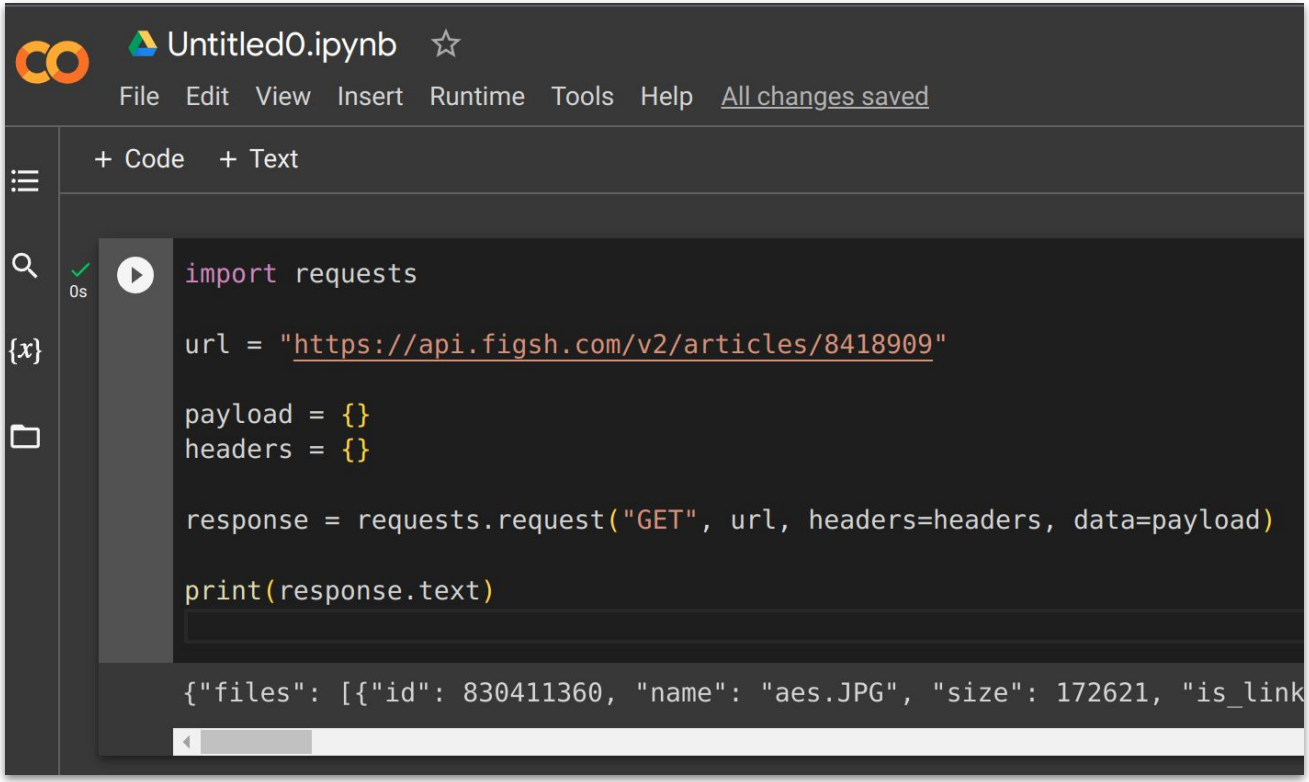
Generate code using Postman

- Postman can generate source code in the language of your choice for the performed requests
- You can easily prototype using a *notebook*, for example https://colab.research.google.com/

Generate code using Postman



Generate code using Postman



Tea Break
30 min

Photo by TeaCora Rooibos on Unsplash

Building an application

Requirements

- Retrieve the items from https://global.figsh.com
- Retrieve the number of views for each item
- Display the data on a HTML page

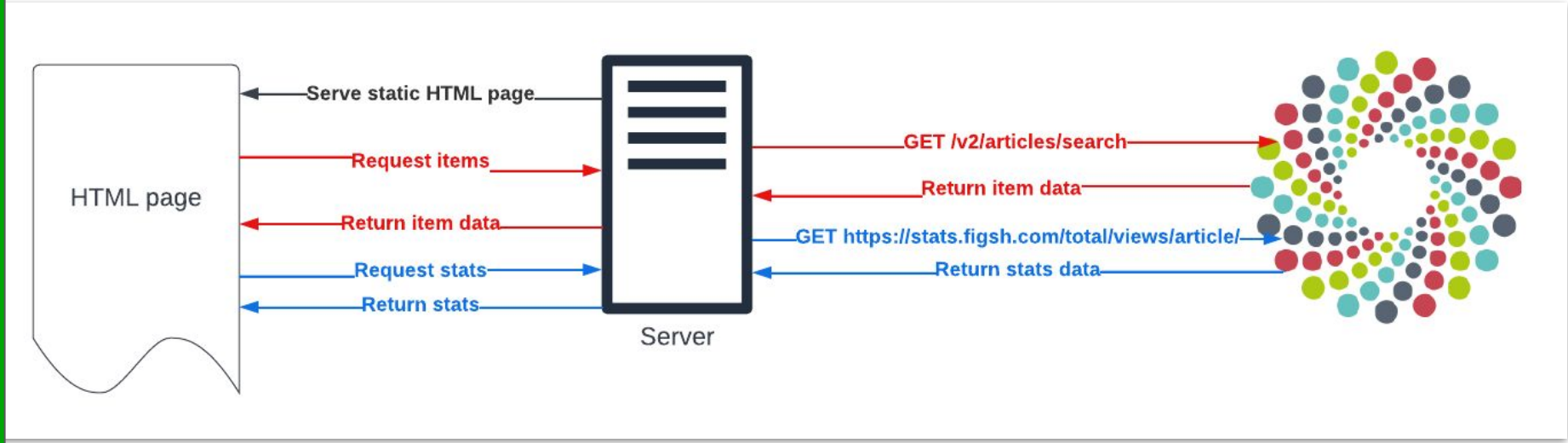
Tools

- Server:
  - Python: https://www.python.org/
  - Flask: https://flask.palletsprojects.com/en/2.3.x/
- Web interface:
  - React: https://react.dev/
- Development environment:
  - Visual Studio Code: https://code.visualstudio.com/
  - Python in Visual Studio Code: https://code.visualstudio.com/docs/python/python-tutorial
- Code generation:
  - ChatGPT: https://chat.openai.com/

CORS

- Abbreviation for cross-origin resource sharing
- Mechanism which indicates from what origins (http://localhost:5000 in our case) a browser is allowed to make HTTP API requests
- Security feature which prevents malicious sites stealing user information from legitimate sites
- By default Figshare's API does not allow cross-origin requests; solutions:
  - Make API requests via a (proxy) server
  - Contact support@figshare.com

System diagram



Further documentation

- Python Flask: https://flask.palletsprojects.com/en/2.3.x/
- Python requests: https://requests.readthedocs.io/en/latest/
- React: https://react.dev/
- Fetch: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API
- Promise: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise

And that’s everything

- What we’ve covered:
  - Basics of HTTP APIs
  - Overview of Figshare’s APIs
  - How to use Postman to perform HTTP requests
  - How to prototype applications that make use of an API
  - How to build a basic application that displays data from Figshare’s API

And that’s everything

- What we’ll make available to you:
  - This presentation: https://docs.google.com/presentation/d/1fTgMwE5c2zuip0B7I7oQgxvAjVkIiFBb1-YTqEYx8XQ/edit?usp=sharing
  - The Postman collection with the performed requests: https://www.postman.com/tudor/workspace/public-workspace/collection/329625-1a755e1d-811a-48b8-8471-860e3f09f0ec
  - The ChatGPT chat: https://chat.openai.com/share/fea4383b-3b32-4608-9699-69aca739aa6f
  - The workshop website: https://amckennafoster.github.io/figshare-api-workshop.github.io/index.html

Q&A & PROJECT TIME

Use this time to ask questions and work on your project.

Thank you!

And don’t hesitate to contact us during OR 2023 and beyond!