The globusworld TOUR

Edge.

Introduction to the Globus Platform and Data Portals

Steve Turoscy – sturoscy@globus.org

May 24, 2023

globus

THE UNIVERSITY OF CHICAGO

Argonne NATIONAL LABORATORY

Globus Platform Services

…Integrate Globus capabilities into your data portal, science gateway, data commons or other research app

Globus PaaS accelerates development

- **Auth**
- **Groups**
- **Transfer**
- **Compute**
- **Search**
- **Timer**
- **Flows**
- **GCS Manager**

- **Globus web app consumes the same public APIs**
- **Resources named by URL (standard REST approach)**
- **Request/response body is JSON**

docs.globus.org/api

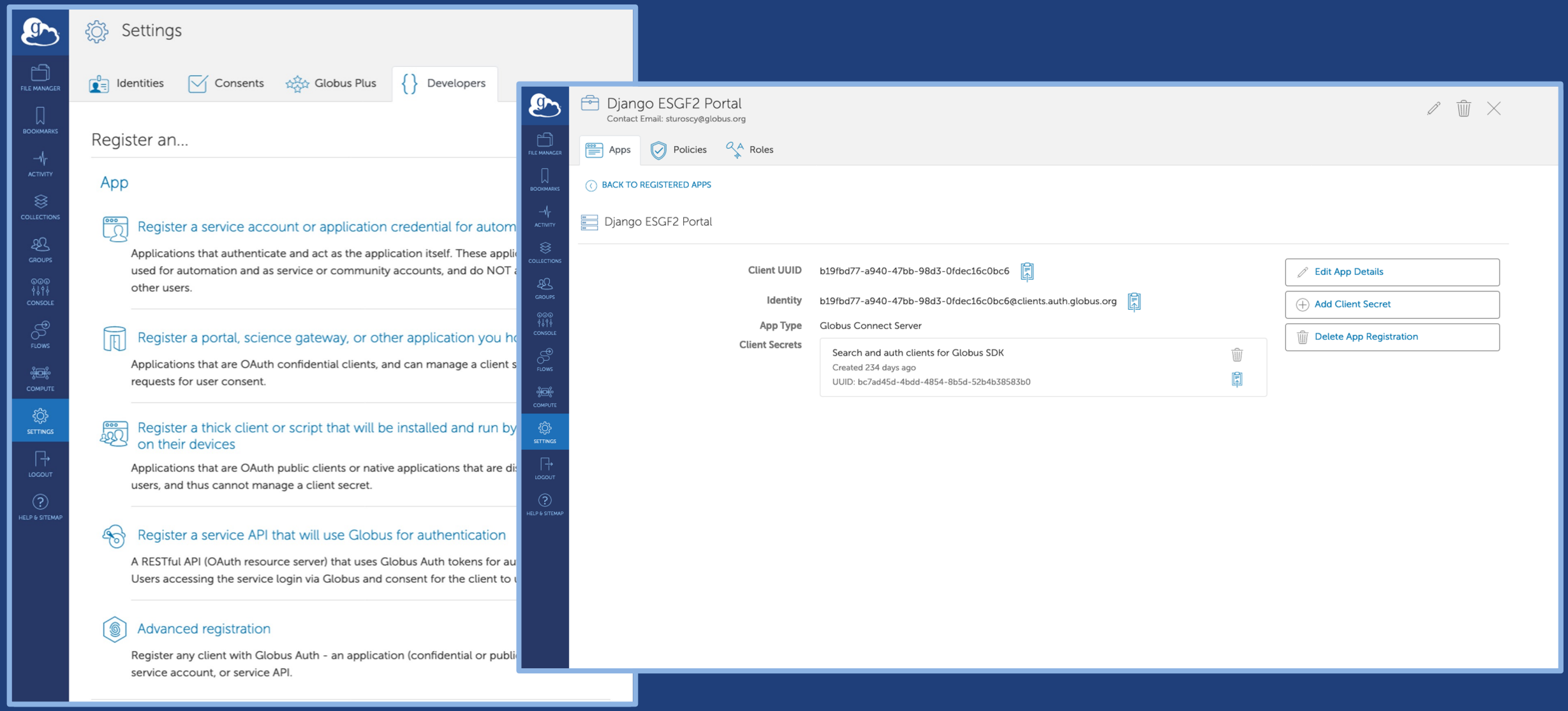
Globus Python SDK

- **Python client library for the Globus REST APIs**
- **Largely direct mapping to REST API**
- `globus_sdk.SearchClient` **class handles index creation and deletion, metadata ingest, and querying**
- `globus_sdk.TransferClient` **class handles connection management, security, framing, marshaling**

globus-sdk-python.readthedocs.io
github.com/globus/globus-sdk-python

App registration

Get app credentials at https://app.globus.org/settings/developers



Globus Auth: Foundational IAM service

https://globus-sdk-python.readthedocs.io/en/stable/services/auth.html

- **Integral in making your portal widely and easily accessible**
- **Brokers authentication and authorization among…**
  - End-users
  - Identity providers: enterprise, external (federated identities)
  - Services: resource servers with REST APIs
  - Apps: web, mobile, desktop, command line clients
  - Services acting as clients to other services
- **OAuth 2.0 Authorization Framework (a.k.a. OAuth2)**
- **OpenID Connect Core 1.0 (a.k.a. OIDC)**

Fundamental Concepts

- **Scopes**
  - APIs that client is requesting access to
  - Service and resources within that service
- **Consents**
  - Authorizes a client to access a service, within limited scope, on the resource owner's behalf
- **Multiple methods for user to grant consent depending on the type of application**

Several authentication modes supported

- **A portal/science/gateway or other application you host**
  - Authorization code grant: authenticate as user identity
  - Browser redirect; auth code returned automatically; tokens stored securely
- **A command-line/thick client or script that is installed and run on the user's device**
  - Native app grant: authenticate as user identity
  - Auth code returned automatically; tokens stored per installation
- **Service account or application credentials for automation**
  - Client credentials grant: authenticate as application identity
  - Client ID and Secret stored securely
- **Application able to manage tokens for offline/long lived tasks**
  - Request refresh tokens in addition to access tokens

Managing service accounts/app credentials

- **Application Identity: appclientid@clients.auth.globus.org**
- **These are confidential apps with client id and secret**
- **Ensure application is on a secure device**
- **Set up policy for rotation of secret**
- **Assign project admins to manage the registration**

Globus Groups: Use groups for authorization

- **Groups allow your portal to automatically manage access/permissions**
  - Grant membership manager role to your application
  - Your web app can add/remove members to grant/remove access
- **Use groups for your application's permissions**
  - Instead of managing a bunch of ACLs in your application, use group membership
  - Lookup membership
    - Check membership to determine permissions
  - Add/remove members
  - Configure policy settings
  - Create/delete groups

globus-sdk-python.readthedocs.io/en/stable/services/groups.html

Guest collections simplify data access, transfer

- **Users don't need local accounts to access data and the portal can act on the collections using its service account**
- **Grant the application Access Manager role**
  - Allows the application to manage permissions on the collection
  - Set for application identity: appclientid@clients.auth.globus.org
- **Grant roles for management of endpoint and tasks**

`TransferClient` low-level calls

- **Thin wrapper around REST API**
  - `post()`, `get()`, `update()`, `delete()`

  `get(path, params=None, headers=None, auth=None, response_class=None)`
    - `path` – path for the request, with or without leading slash
    - `params` – dict to be encoded as a query string
    - `headers` – dict of HTTP headers to add to the request
    - `response_class` – class response object, overrides the client's `default_response_class`
    - Returns: `GlobusHTTPResponse` object

globus-sdk-python.readthedocs.io/en/stable/services/transfer.html

`TransferClient` higher-level calls

- **One method for each API resource and HTTP verb**
- **Largely direct mapping to REST API**

```
endpoint_search(filter_fulltext=None,
                filter_scope=None,
                num_results=25,
                **params)
```

Getting tokens: jupyter.demo.globus.org

- Sign in with Globus and verify the consents
- Start My Server and open `globus-jupyter-notebooks`
- Run `Platform_Introduction_Native_App_Auth`
- Alternatively, use notebooks on your own machine:

github.com/globus/globus-jupyter-notebooks

Globus Search and the Django Globus Portal Framework

https://github.com/globus/django-globus-portal-framework

Making Data Findable with Globus Search

Data description and discovery

- **Metadata store with fine-grained visibility controls**
- **Schema agnostic dynamic schemas**
- **Simple search using URL query parameters**
- **Complex search using search request document**

docs.globus.org/api/search

Data ingest with Globus Search

POST /index/{index_id}/ingest'

```
{
  "ingest_type": "GMetaList",
  "ingest_data": {
  "gmeta": [
    {
      "id": "filetype",
      "subject": "https://search.api.globus.org/abc.txt",
      "visible_to": ["public"],
      "content": {
        "metadata-schema/file#type": "file"
      }
    },
    ...
  ]
}
```

- Bulk create and update
- Task model for ingest at scale

Data ingest with Globus Search

POST /index/{index_id}/ingest'

```
{
  "ingest_type": "GMetaList",
  "ingest_data": {
  "gmeta": [
    {
      "id": ”weight",
      "subject": "https://search.api.globus.org/abc.txt",
      "visible_to": ["urn:globus:auth:identity:46bd0f56-
                     e24f-11e5-a510-131bef46955c"],
      "content": {
        "metadata-schema/file#size": ”37.6",
        "metadata-schema/file#size_human": ”<50lb”
      }
    },
    ...
  ]
}
```

Visibility limited to Globus Auth identity
- Single user
- Globus Group
- Registered client application

Data discovery with Globus Search

GET /index/{index_id}/search?q=type%3Ahdf5

```
{
  "@datatype": "GSearchResult",
  "@version": "2017-09-01",
  "count": 1,
  "gmeta": [
    {
      "@datatype": "GMetaResult",
      "@version": "2019-08-27",
      "entries": [
        { ... }
      ],
      "subject": "https://..."
    }
  ],
  "offset": 0,
  "total": 1
}
```

Simple query

Data discovery with Globus Search

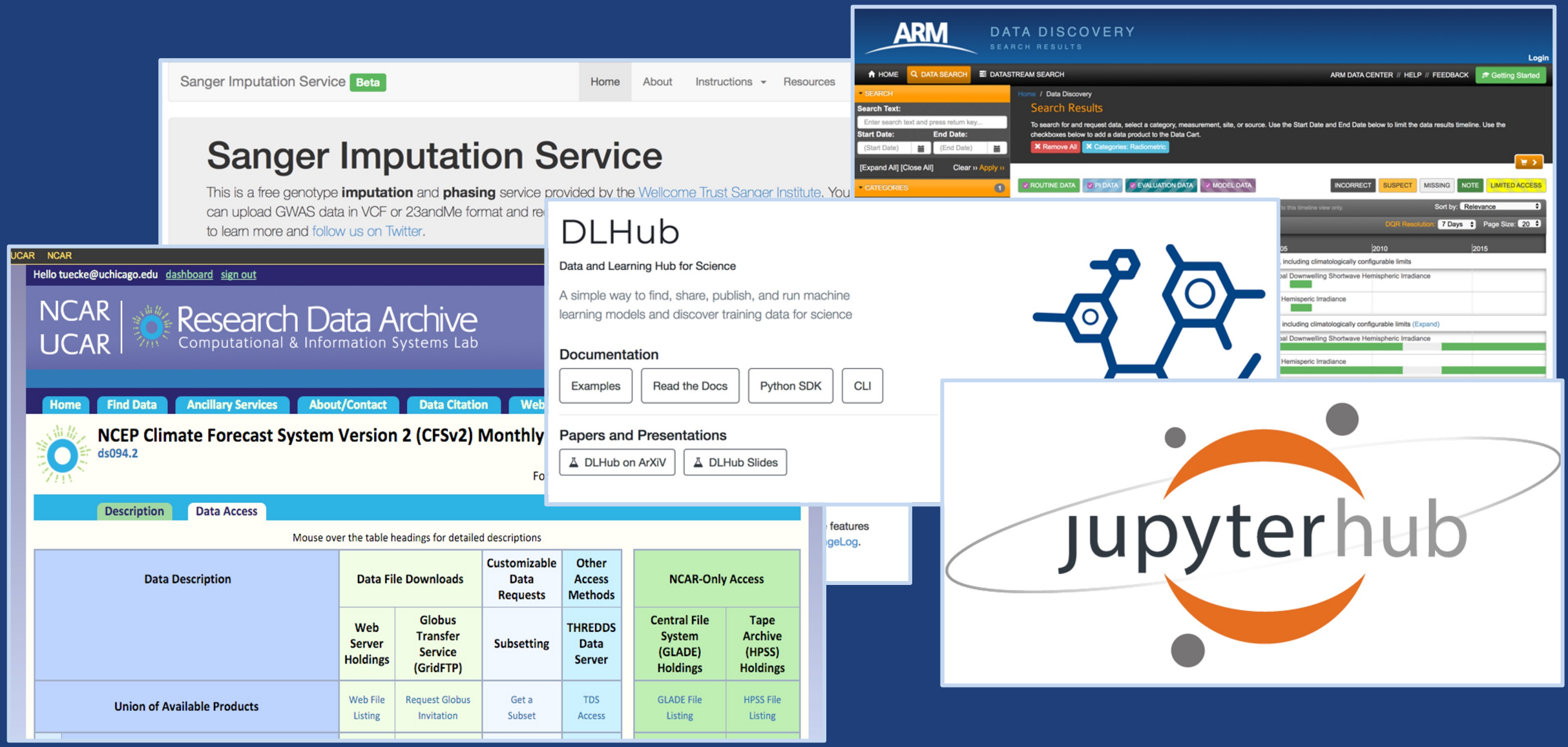
POST /index/{index_id}/search

```
{
  "filters": [
    {
      "type": "range",
      "field_name": ”pubdate",
      "values": [
        {
          "from": "*",
          "to": "2020-12-31"
        }
      ]
    }
  ],
  "facets": [
    {
      "name": "Publication Date",
      "field_name": "pubdate",
      ...
    }
  ]
}
```

Complex query

- Filter
- Facets
- Boosts
- Sort
- Limit

Example portals/science gateways/apps



Evolving the MRDP design pattern

Input form

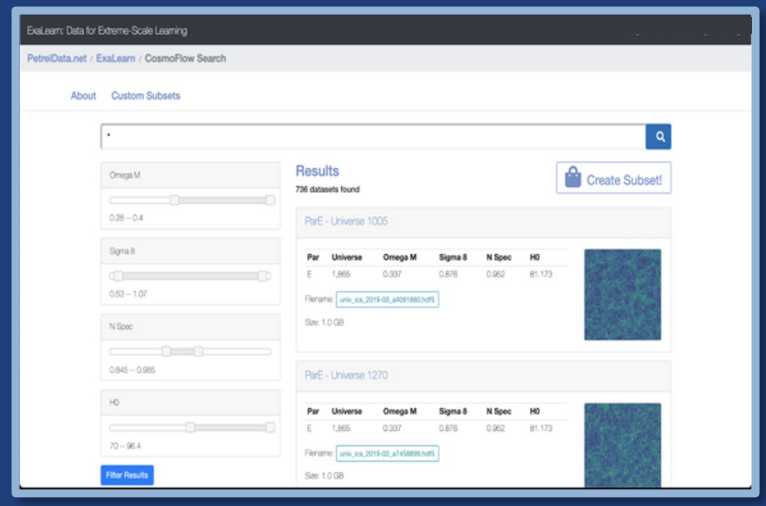
Automated Extraction

Bulk ingest

Globus Search

Ingest metadata, set visibility policies

MRDP



Enabling discoverability: MRDP + Faceted Search

Portal Core Functionality

- **User authentication**
- **Django-based framework**
  - Portal URL mappings
  - Token loading
- **Service calls to Globus Search**
- **Manage request lifecycle**
- **Post process search requests**

User authentication

- **Scopes are configured in the portal**
- **Users authenticate with Globus using standard flow**
  - Python Social Auth used for Authentication backend
- **User tokens are saved in the database**
- **Future requests authorized with user access tokens**
  - Searches use Search bearer token

```
SOCIAL_AUTH_GLOBUS_SCOPE = [
    'urn:globus:auth:scope:search.api.globus.org:all',
    'urn:globus:auth:scope:transfer.api.globus.org:all',
    'urn:globus:auth:scope:groups.api.globus.org:view_my_groups_and_memberships',
```

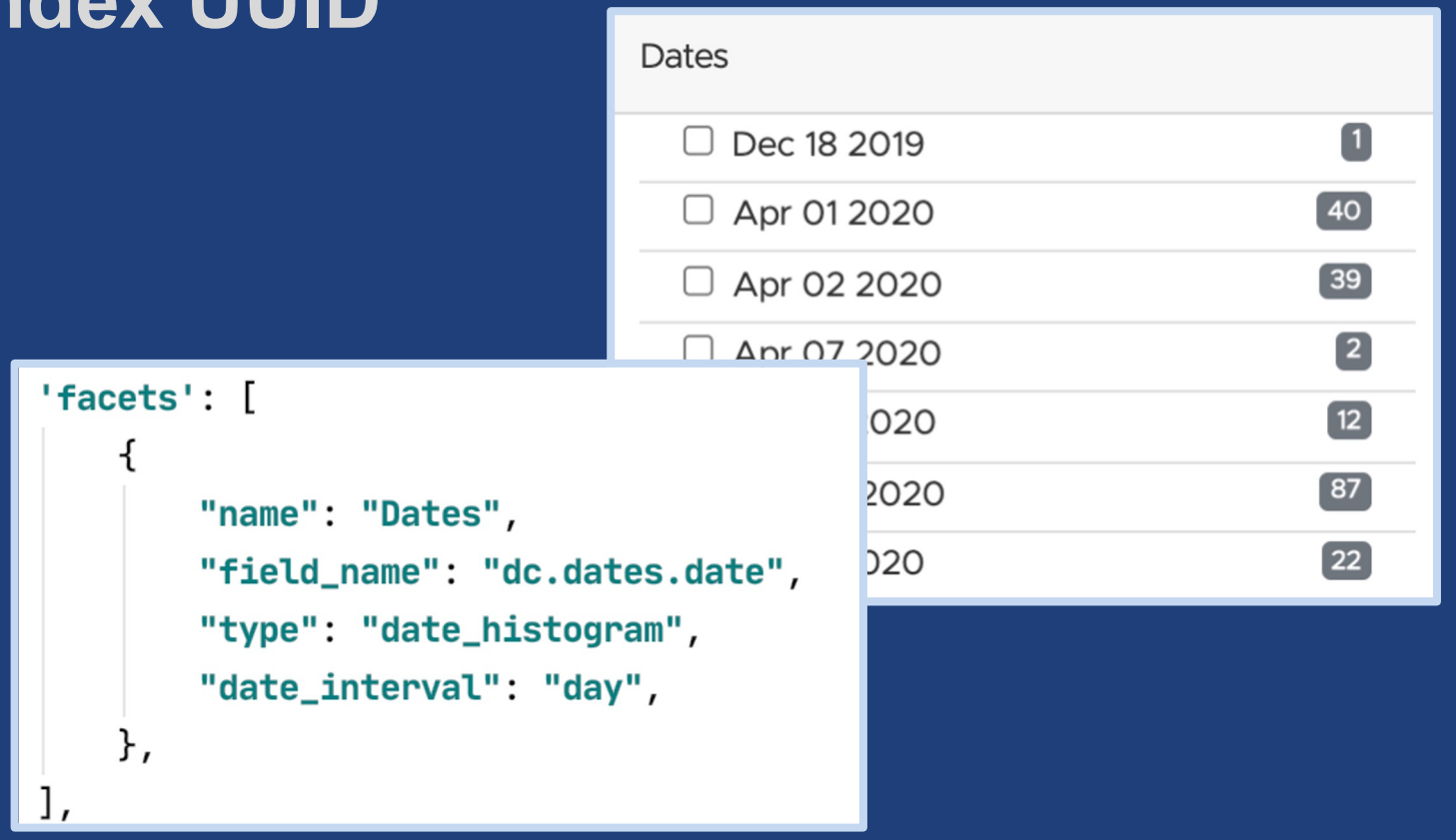
Portal service calls use the Globus SDK

- **Globus portal framework loads tokens from database**
- **Globus service object instantiated with token**
- **Call to Globus service(s)**
- **Portal renders result in templates**

```
def list_my_files(request):
    tc = load_transfer_client(request.user)
    path = tc.operation_ls(request.GET['endpoint'], path=request.GET['path'])
    return render(request, 'my_view.html', {'dir': path.data}
```

An index is configuration driven

- A Search index is configured in portal settings
- Add Globus Search index UUID
- Add a name
- Add facets
- Add fields
- Start searching!



Lifecycle of a request

- **User makes a query**
- **Portal sends request to Globus Search**
  - Request contains user bearer token
- **Portal receives response**
- **Portal does processing on response**
  - Parse Dates, build URL for Globus webapp, etc.
- **Portal renders data into templates**
- **User receives a search page**

Creating your science gateway using the Globus portal framework

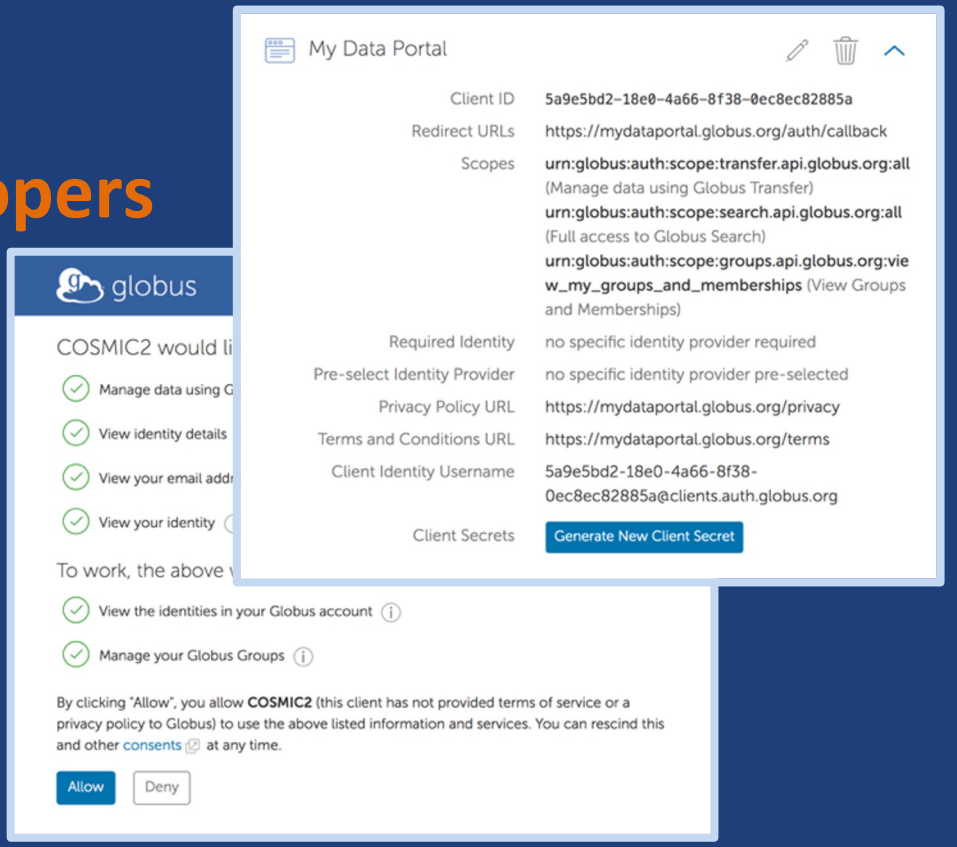
https://bit.ly/gw-tut

Source: github.com/globus/django-globus-portal-framework

Docs: django-globus-portal-framework.readthedocs.io/en/stable/

Application registration

https://app.globus.org/settings/developers



- **Set redirect URLs**
- **Get client ID and secret**
- **Consents implement least privileges principle**

Redirect URLs

`https://tutN.globusdemo.org:8443/`
`https://tutN.globusdemo.org:8443/complete/globus/`

Portal deployment

- **Install dependent libraries**
  - For production use, add robust WSGI/ASGI server
- **Deploy a portal instance using** `cookiecutter`
- **Configure settings**
- **Run and use!**
- **Future: containers**

Exporting data via Globus
from easy to custom

Where's the data?

- **Remember – we're using Globus Connect, so your datasets are in a Globus collection**
- **Three options for enabling transfers from your portal:**
  1. Link to the collection in the Globus web app (Easy! But not customizable.)
  2. Use the Globus Helper Page (Easy! A bit customizable.)
  3. Use a JavaScript interface (Less easy. Very customizable.)
- **Let's see an example of each…**

Easy: Link to the Globus web app

Easy: Select destination with Globus Helper Page

Bulk Transfer via Globus Helper Pages

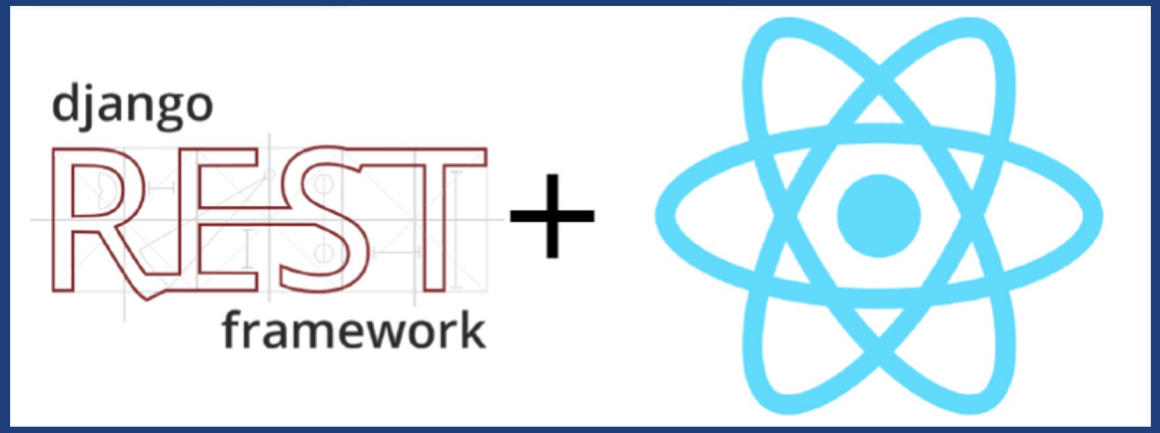
Even though this is a little more complicated and requires some amount of additional setup, you get the benefit of using the Globus Web App to handle authorization as well as being able to customize your portal a little further

docs.globus.org/api/helper-pages/browse-endpoint

Advanced: Create a custom UI that uses Globus SDK

Advanced Customization via Django REST Framework and Javascript

More complicated!



Why?

- Complete control over form and function
- Benefits of a Single Page Application (SPA)

- www.django-rest-framework.org
- reactjs.org
  - recoiljs.org/ (manage state)
  - reactrouter.com/en/main
- webpack.js.org

Support resources

- Globus documentation: docs.globus.org
- GitHub: github.com/globus
- Migrating applications to use GCSsv5: docs.globus.org/globus-connect-server/migrating-to-v5.4/application-migration/
- Sample code: github.com/globus/automation-examples
- YouTube channel: youtube.com/user/GlobusOnline