Integrating ESAP with the Zooniverse

ESAP Tech Talk - 21-09-20

Hugh Dickinson

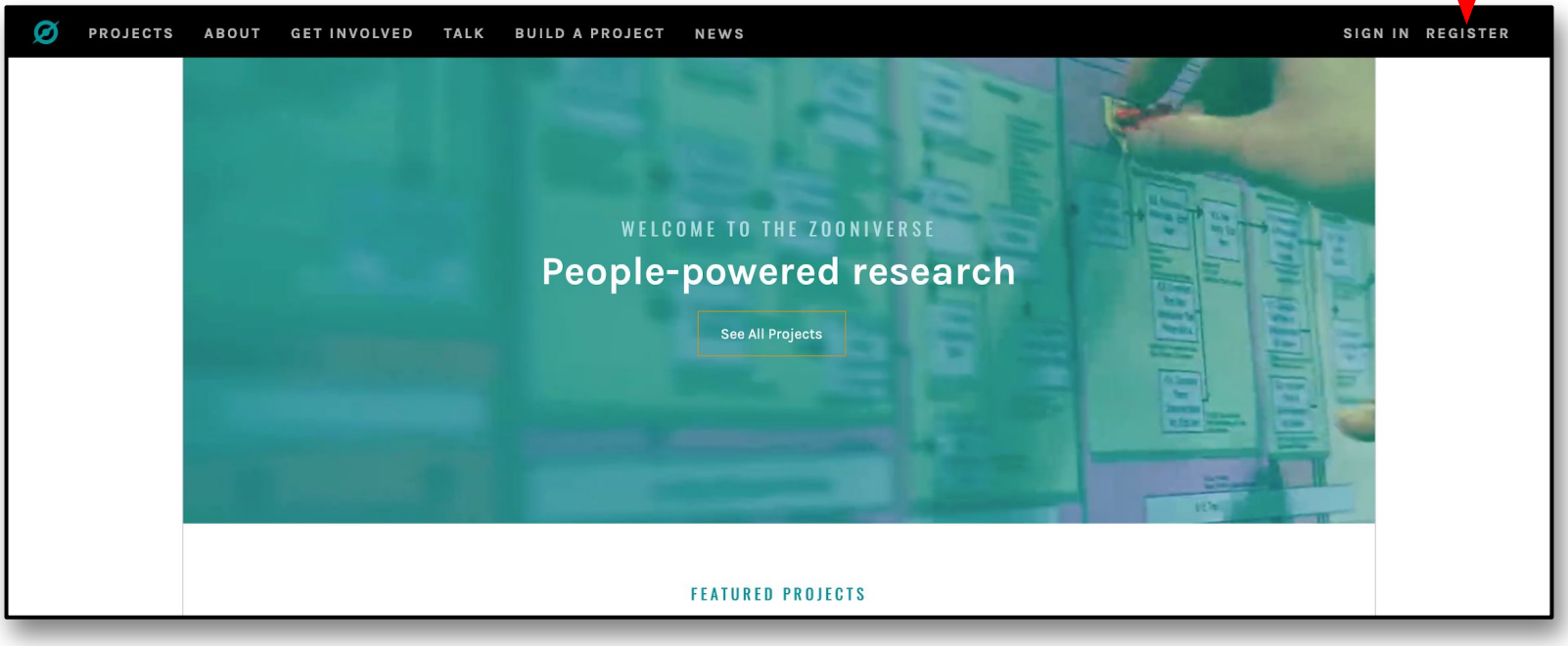
Outline

1. Zooni-what?
2. Zooniverse and ESAP - Quick Demo.
3. Architectural Overview.
4. The Backend.
5. The Frontend.
6. What’s next?

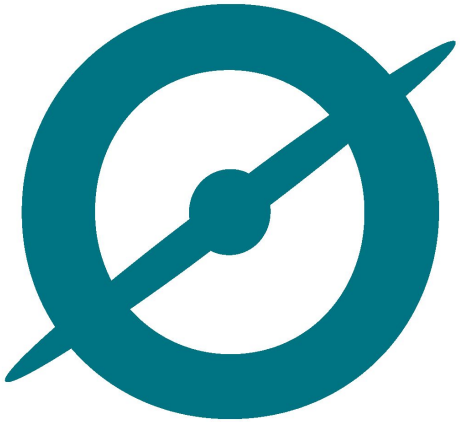
The Zooniverse

zooniverse.org

Register Here



Zooniverse and ESAP

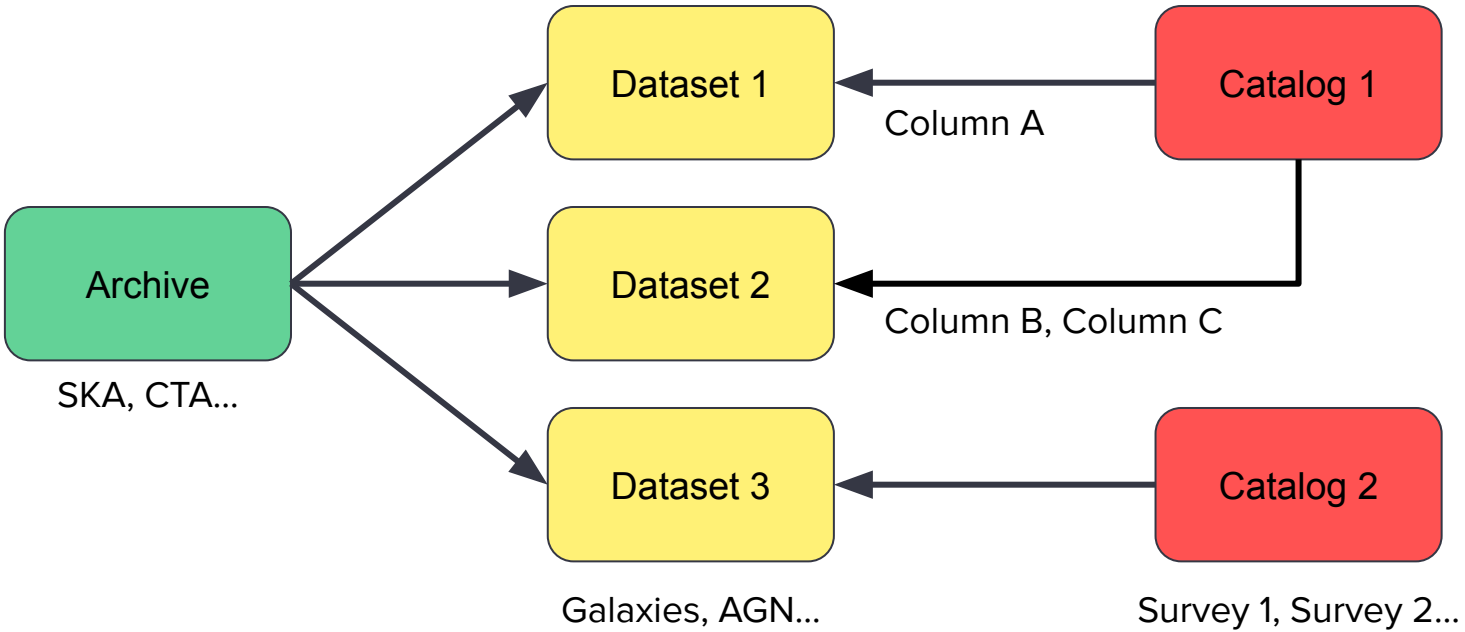


Quick Demo

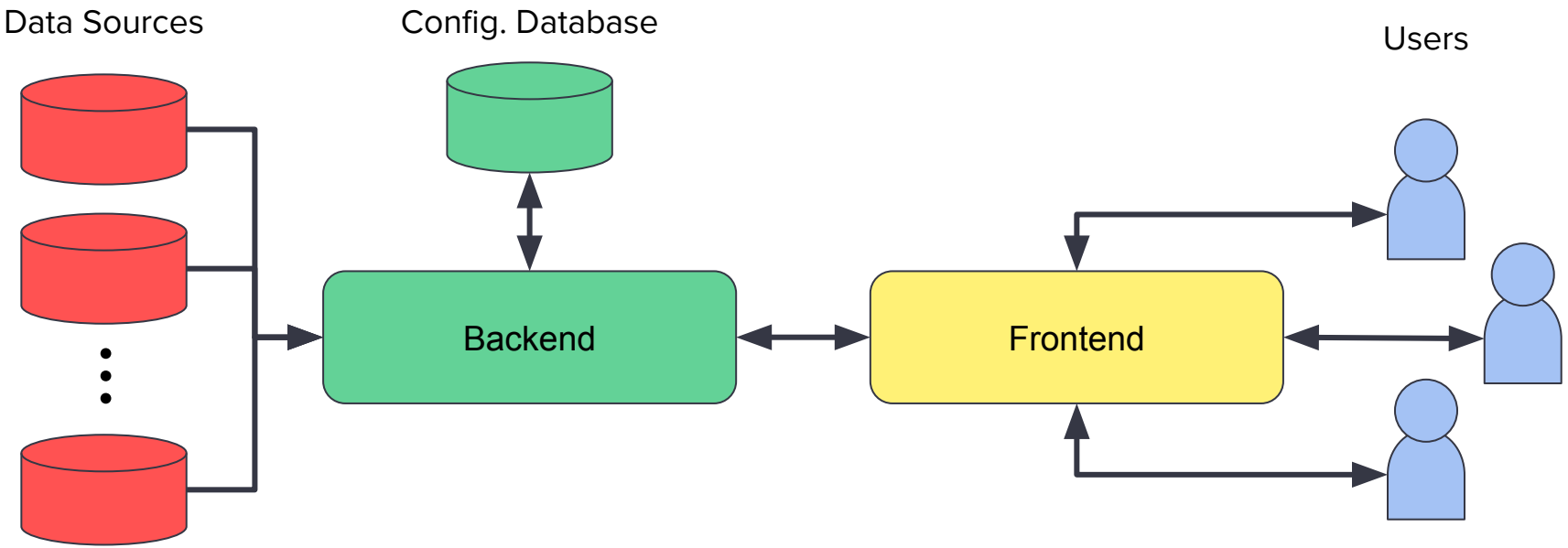
Data Discovery Data Model

- **Three** types of entity
  - ***Archives*** - Intended to encapsulate **many data sources** associated with a **single observatory** or facility e.g. SKA, CTA, LHC
  - ***Catalogues*** - Intended to encapsulate **concrete data resources** e.g. data tables, imaging catalogues.
  - ***Datasets*** - An abstract link between catalogues with archives.
    - Many datasets can reference the **same catalogue**, but may access different components e.g. different columns of a table.
    - Datasets belong **uniquely** to a **single archive**.

ESAP *Data Discovery* Data Model Schematic



The Frontend and the Backend



The Backend

https://git.astron.nl/astron-sdc/esap-api-gateway

- The ESAP backend is written in **Python** using the **Django** framework.
- Development implementation uses a SQLite3 database.
- Implements connection with **Virtual Observatory** (VO) services “out of the box”.
  - If your data source provides a VO interface, you may not have any integration work to do at all!
- In general, you must **implement two Python files, update one Python file** and **update the backend database** to integrate your data source.

The Backend - The Database

- Navigate to: **`<host>:<port>/esap-api/admin/query`**
- Add a new ***Archive***.
- Add one or more ***Catalogue***s.
- Define one or more ***Dataset***s.
- **Make a note of**:
  - Dataset **service connector** module/class names.
  - **URIs** for archives and catalogues.

The Backend - Configuration

- Single file provides frontend UI definition per `react-jsonschema-form`.
  - https://react-jsonschema-form.readthedocs.io/en/latest/
- Also defines GUI **name** (make a note of this), **appearance**, and **navigation** options. **Two data structures** specify the form structure:
  - `query_schema` - Basic form structure (required)
  - `ui_schema` - Advanced form field properties and validation requirements (optional)
- **Dynamic field discovery** is possible.
- Zooniverse examples:
  - https://git.astron.nl/astron-sdc/esap-api-gateway/-/blob/master/esap/esap/configuration/zooniverse.py
  - https://git.astron.nl/astron-sdc/esap-api-gateway/-/blob/master/esap/esap/configuration/zooniverse_fields.py

The Backend - The Service Connectors

- One Python file per ***Dataset***, implementing the ***Service Connector*** class specified in the database.
- Two required methods:
  - `construct_query`
    - Returns a list of query strings that can be interpreted by `run_query`.
  - `run_query`
    - Receives the list of query strings constructed by `construct_query` and retrieves the requested data.
- Zooniverse example:
  - https://git.astron.nl/astron-sdc/esap-api-gateway/-/blob/esap-gateway-zooniverse/esap/query/api/services/zooniverse.py

The Backend - The Service Connectors

- Finally, to make sure your service connector is **registered with the backend**, add it to `instantiate_connector` in `query_controller.py`.
- Zooniverse example:
  - https://git.astron.nl/astron-sdc/esap-api-gateway/-/blob/esap-gateway-zooniverse/esap/query/api/services/query_controller.py#L13
- **For some data sources** you may need to consider implementing customized **pagination** and ***most*** will require default **serialization** specifications.
- The **Zooniverse** integration **overrides** the **default implementations** for both pagination and serialization.
- This is required to enable dynamic specification of fields for retrieval.

The Backend - Serialization

- The Backend uses the **Django REST Framework** Serializers module:
  - https://www.django-rest-framework.org/api-guide/serializers/
- To support serialization your service connector class ***must*** implement a nested `CreateAndRunQuerySerializer` class.
- For queries that return a **static set of fields** this class is trivial to implement - the field names and types are defined as **class attributes**.
- The Zooniverse connector supports **dynamic field selection**, so the serializer implementation is more complicated - **fields must be defined on instantiation**.
- Zooniverse example:
  - https://git.astron.nl/astron-sdc/esap-api-gateway/-/blob/esap-gateway-zooniverse/esap/query/api/services/zooniverse.py

The Backend - Pagination

- The Backend uses the **Django REST Framework** Pagination module:
  - https://www.django-rest-framework.org/api-guide/pagination/#pagination
- ***Bear in mind*** that using the default pagination scheme **may** involve the backend **retrieving the entire query result** then paginating it locally.
- Your data source may implement its own pagination and it may be **much more efficient** to request **single pages** as required.
- **Default pagination can be disabled** using connector class attribute or using a query parameter from the front-end.
- **If you disable default pagination**, your query response ***must*** include:
  - `requested_page` - ***integer*** - the page being returned.
  - `pages` - ***integer*** - the total number of pages that are available

Questions so far?

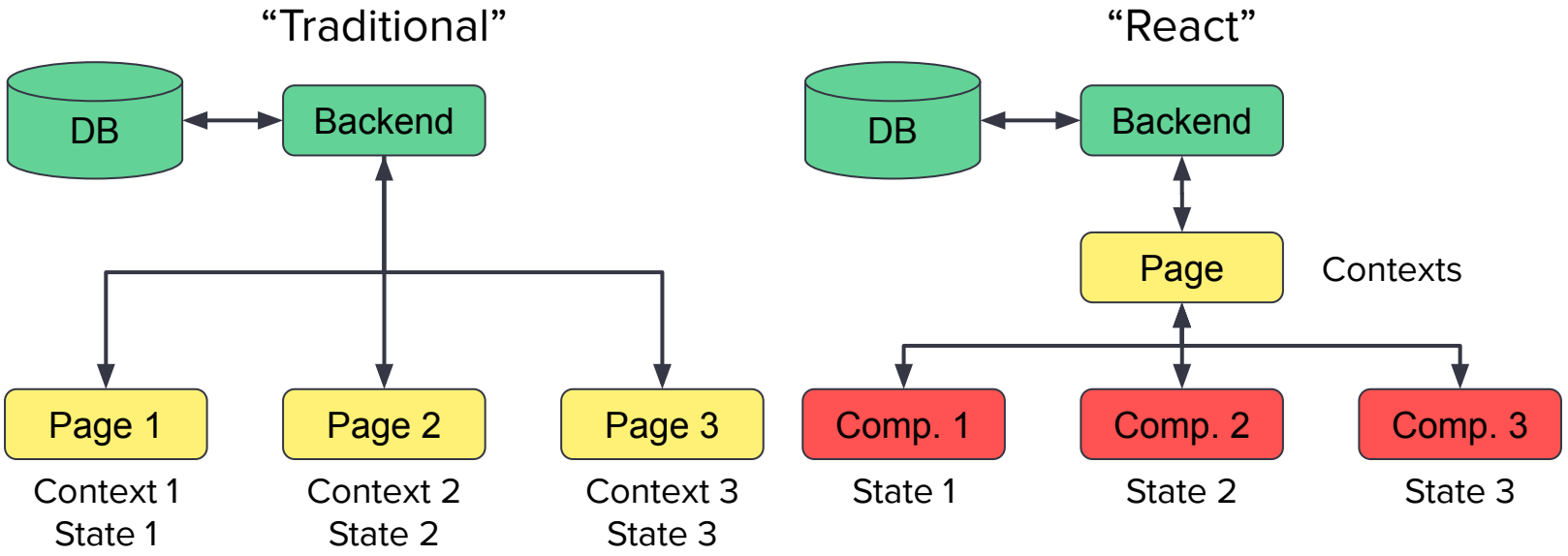
The Frontend

https://git.astron.nl/astron-sdc/esap-gui

- The ESAP frontend is written using the **JavaScript *React*** framework.
  - https://reactjs.org
- ***Rendering and Layout*** uses the ***React Bootstrap*** framework.
  - https://react-bootstrap.github.io
- In general, you must **implement two JavaScript files** and **update one JavaScript file** to integrate your data source.
- You ***may*** wish to define additional CSS styles in
  - https://git.astron.nl/astron-sdc/esap-gui/-/blob/esap-gui-dev/src/App.css

The Frontend - Behind the Scenes

- **Simple *React*** apps use a single page with shared context objects. *React* manages **rendering** and **updates** of **components**.



The Frontend - Parsing the query form

- **Update** `parseQueryForm.js` - This is where you need the **query schema name** you noted down!
  - https://git.astron.nl/astron-sdc/esap-gui/-/blob/master/src/utils/form/parseQueryForm.js
- **Implement** a **function to parse data from the form** and assemble a query for the backend.
- Zooniverse implementation is `parseZooniverseForm.js`.
  - Determines **which catalogue was selected** and prepares a query accordingly.
  - **Returns a list** containing (in this case) a **single query string**.
- Zooniverse example:
  - https://git.astron.nl/astron-sdc/esap-gui/-/blob/master/src/utils/form/parseZooniverseForm.js

The Frontend - Rendering Query Results

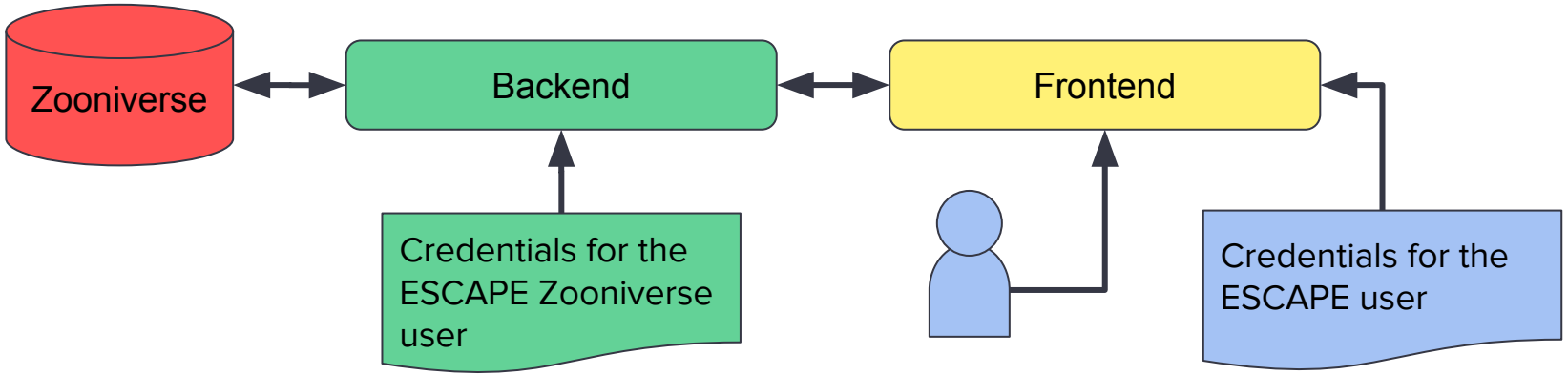
- **Update** `QueryCatalogs.js` - This is where you need the **archive URI** you noted down!
  - https://git.astron.nl/astron-sdc/esap-gui/-/blob/master/src/components/query/QueryCatalogs.js#L22
- **Update** `QueryResults.js` - This is where you need the **catalogue URIs** you noted down!
  - https://git.astron.nl/astron-sdc/esap-gui/-/blob/master/src/components/query/QueryResults.js
- **Implement** a **component to render the query results** for your data source.
- Zooniverse example:
  - https://git.astron.nl/astron-sdc/esap-gui/-/blob/master/src/components/query/ZooniverseResults.js

The Frontend - `ZooniverseResults.js` Walkthrough

- Base component - `ZooniverseResults`
  - Obtains a reference to the `QueryContext`.
  - Instantiaties **specific components** depending on catalogue URI.
- Separate components for each catalogue preprocess data and compute fields to render.
- `ZooniverseProjectResults` renders a **simple table**.
- `ZooniverseWorkflowResults` renders a **nested table**.
- **Simple pagination callback** invokes `setPage()` obtained from `QueryContext`.

Coming soon!

- **Adding classification data** that you discover to your **shopping cart**.
- **Analysing classification data** interactively using custom **Jupyter Notebooks**.
- **Generating new projects** using your data from the ESAP interface
- **Access private projects by granting** access rights to the ***ESCAPE* Zooniverse user**.



Questions now?