A. DPLA Phase 2 Ingestion Scope of Work

Introduction

The deliverables in this track are focused to create a GUI “dashboard” web application to allow our non-technical staff to queue and execute harvests, mappings, enrichments, and indexing. The basics needed for such a system are described in very high-level requirements below. We have begun to think through some of the details of how this system will work through the work we have done so far on the individual pieces, but in order to truly create a working system in the time frame we need we are looking for additional help.

We may be able to execute some of the basic work on these requirements before work by a contractor would commence, at which point we would adjust the specific deliverables. However, we cannot predict at this point what we would be able to achieve and we would be very happy to work with you on developing this further as needed.

Deliverable:
A Rails-based “dashboard” application that implements all of the functions indicated in the draft requirements, integrated with Krikri. The application may extend or replace the current Heidrun application.

Requirements

- **Status pages**
    - Listing statuses of most recent activities
        - For all providers (access restricted to administrators)
        - For individual providers (access restricted by provider user accounts)
        - With sensible defaults and controllable parameters for how many results to display (e.g. activities updated in last n days plus those still running)
        - Showing for each activity (see "Preparation" below)
            - Running / Complete / Failed / Halted status
            - Statistics
                - Number of log messages with message severity level
                    - Where aggregate stats are probably kept in the existing table for activities in the Postgresql `ingestion` database.
                    - With link to central logging system (e.g. Kibana), filtered on the activity

            - (Bonus) Records processed. If total record count can be known (in the case of a mapping, enrichment, or indexing, by querying the generator activity), show the estimated time to completion.
        - (Bonus) Button to stop a running activity, with modal window confirmation. (See "Preparation" below.)
    - With links to a control-panel page for each provider, described below in "provider controls"
- **Provider controls** (a page for each provider)
    - Manage Provider Profile
        - Harvest source
            - Type (class)
            - URL or file, depending on the type
            - Options consisting of pairs of "key" and "value" input boxes
                - "add" and "delete" buttons to add or delete pairs
        - Enrichment Profiles
            - List Enrichment Profiles for a Provider
            - One Enrichment Profile is the default
            - For each Enrichment Profile
                - It is a list of Krikri::Enrichments classes
                - You can select an enrichment from a dropdown to add it to the list.
            - The user can order Enrichments within an Enrichment Profile
        - Mapping
            - (restricted to admins) Select whether to allow file uploads. See "Prep" item below.
            - File upload. If enabled, directory gets an application-assigned location based on provider name.
        - QA index name (restricted to admins)
    - Enqueue a harvest with the existing harvest source in the profile (above).
        - Basically a big "go" button with some kind of confirmation.
    - Enqueue a mapping with the existing mapping data in the profile (above).
        - Select the generator harvest activity upon which to base the mapping
    - Enqueue an enrichment
        - Select the enrichment chain (there being perhaps more than one per provider) where the default is automatically selected.
        - Select the generator mapping activity upon which to base the enrichment.
    - Enqueue an indexing
        - **One or the other** of (depending on state of Elasticsearch QA index Research & Development)
            - Select from a dropdown of destination index names
            - Select "QA" or "Production"
        - And confirm whether to proceed in a modal window.

More complete documentation is available:

- Requirements
- Usage Scenarios (draft document)
- Personas (draft document)

Preparation

Potential prerequisites for the features above are given below. Unless otherwise specified, a contractor will take on this work.

- Create classes, data model, REST endpoints for
  - Activities (including stats data structure with properties for error counts and records processed)
  - Provider profiles
  - Enrichment chains
  - Mapping uploads and mapping file search path
    - Add mapping file search path to Krikri. May contain multiple directories. Git working copies can coexist with file-upload directories. See "Provider controls" mapping section above. Given a name "some_provider", the path is searched to find the mapping as "some_provider.rb". Providers should not be able to upload files through the webapp to git working copies -- it should be either / or.
- Add notion of status to Activity.
  - Add classes
    - Krikri::JobStatus::Running
    - Krikri::JobStatus::Stopped
      - Krikri::JobStatus::Halted < Krikri::JobStatus::Stopped
      - Krikri::JobStatus::Failed < Krikri::JobStatus::Stopped
      - Krikri::JobStatus::Complete < Krikri::JobStatus::Stopped
  - In exception handling or job completion, set these statuses
  - Figure out some way to stop jobs programmatically so that Halted can be registered.
- Add aggregate error statistics to Activities
- ~~Add at least a basic central logging system that can be linked to from an Activity's statistics display, with the Activity's ID.~~ **(DPLA is working on this item)**

Not Included

- Plans (scheduled series of Activities). We can add Plans later to the things that you can enqueue in "provider controls." Let's assume we'll always allow users to enqueue one-off activities without having to create a Plan, even after we implement this functionality.
  - The original draft scope of work mentions "auto-indexing" after the mapping and enrichment activities, which requires a complete Plan implementation. This has been left out of the scope above.

- The original draft scope of work lists "Reporting when completed" under each enqueuing category. We assume, though, that the status indicators included in the “Status Pages” described above will serve to report on activities’ completion. Email alerts may be a good feature, but we assume that we will scope those separately for completion by DPLA developers, as they apply to commandline-enqueued activities in addition to ones run through the user interface that is being addressed here.