Brookhaven National Laboratory

U.S. DEPARTMENT OF ENERGY

NSR Workflow

Benjamin Shu
National Nuclear Data Center

October 30th, 2025

@BrookhavenLab

Effects of Current Events

- Budget constraints
  - No contractors for NSR as of April 2025
  - Reduced ability to add/update entries

Effects of Current Events

- Budget constraints
  - No contractors for NSR as of April 2025
  - Reduced ability to add/update entries
- Shifting focus towards AI-readiness
  - Like it or not, generative AI is here to stay

Effects of Current Events

- Budget constraints
  - No contractors for NSR as of April 2025
  - Reduced ability to add/update entries
- Shifting focus towards AI-readiness
  - Like it or not, generative AI is here to stay
- **NSR's format, data model, and workflow need to adapt**
  - Updating the workflow gives us opportunities to leverage AI
  - Updating the data model makes NSR more AI-friendly

Exchange Format

- *Not used anywhere other than NSR*
  - Requires custom parsing code to read
  - Makes processing needlessly difficult

```
<KEYNO   > 2023WIZZ                Unique ID
<HISTORY > A20230605               Entry/modification dates
<CODEN   > CONF Sacramento...      Source type and information
<REFRENCE> Proc.15th.Intern...     Citation information
<AUTHORS > D.Wiarda, ...           List of authors
<TITLE   > Modernization...        Formatted title
<KEYWORDS> [N/A]                   Sentences about content
<SELECTRS> [N/A]                   Structured search keys
```

Exchange Format (contd.)

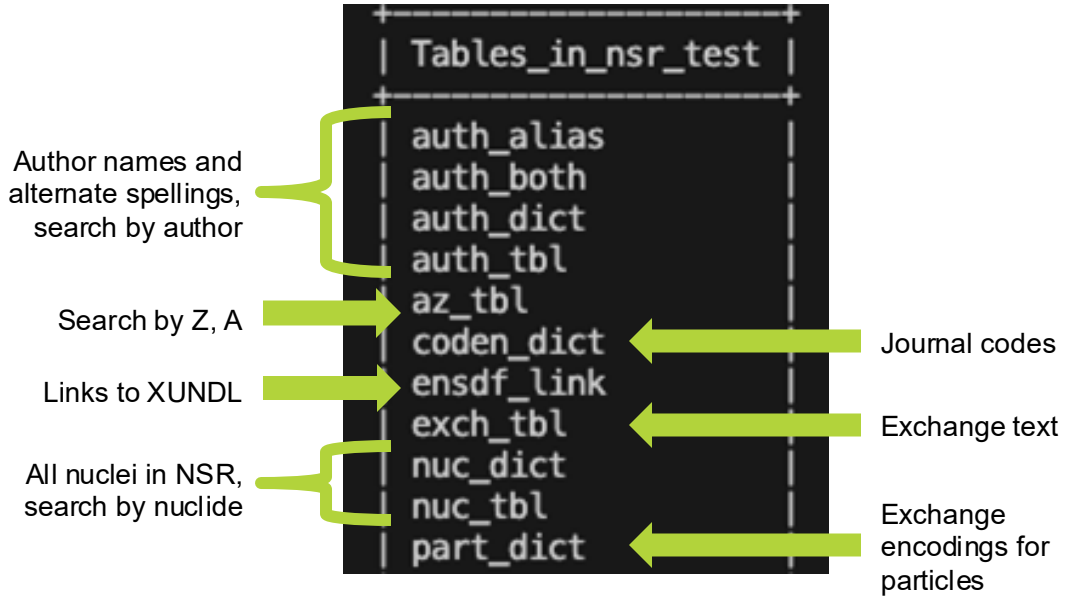
- Still uses "fake LaTeX" for Greek/math symbols
  - *Not used by any other database*
  - Requires further translation after reading exchange format

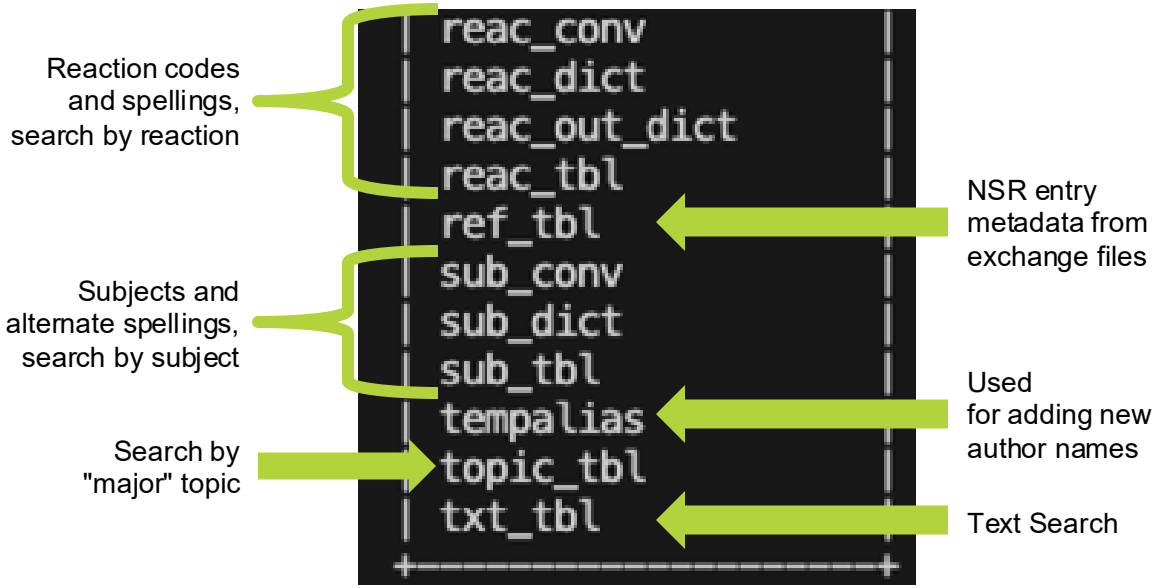
```
<KEYWORDS>RADIOACTIVITY {+37},{+38}Al(|b{+-}),(|b{+-}n)[from {+48}Ca({+9}Be,X),E
=140 MeV/nucleon]; measured  E|g, I|g, I|b, |g|g-coin, |b|g-coin; deduced T{-1/2
}. {+37}Si; deduced levels, J, |p, T{-1/2} of isomeric states at 68- and 156-keV
, B(M1), matrix elements for low-lying ground-state M1 transitions. {+36},{+38}S
i; deduced E|g, I|g, transition intensities. Comparison to other experimental da
ta and shell model calculations using the SDPF-MU, SDPF-U-SI, and FSU interactio
ns. CeBr{-3} based implantation detector with position-sensitive photomultiplier
 tube surrounded by 16 segmented Ge detectors (SeGa) and 15 LaBr{-3} detectors a
t Coupled Cyclotron Facility (NSCL, MSU)
```

https://www.nndc.bnl.gov/nsr/?search-type=keynumber&key=2023OG04

Exchange "Data Model"

- NSR's exchange format *does not store all of its data*
  - Searching requires SQL tables of dictionaries and links

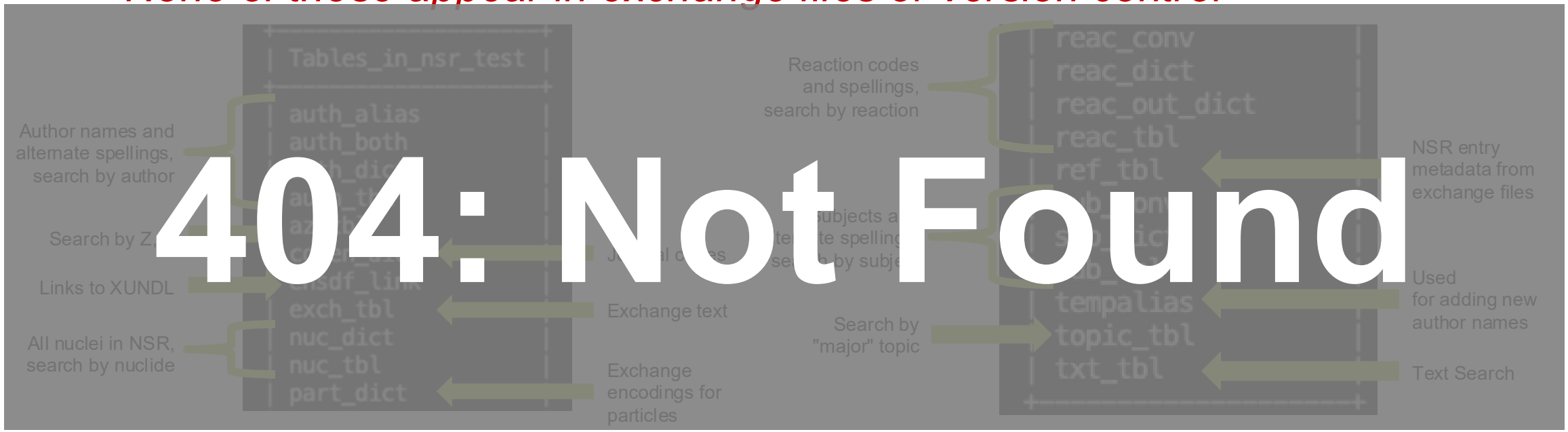




Brookhaven National Laboratory

Exchange "Data Model"

- NSR's exchange format *does not store all of its data*
  - Searching requires SQL tables of dictionaries and links
  - *None of these appear in exchange files or version control*



404: Not Found

Brookhaven National Laboratory

A Better Data Model...

- ...would define *every* database component using consistent structures and formats
  - NSR's current model (exchange + SQL tables) does not do this!

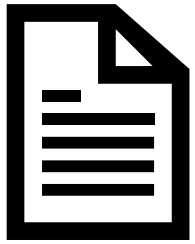
A Better Data Model...

- ...would define *every* database component using consistent structures and formats
  - NSR's current model (exchange + SQL tables) does not do this!

- Changing the data model:
  - Enables translation while preserving original data
  - Greatly simplifies parsing
  - Makes NSR easier for AI to ingest

A Data Model for NSR

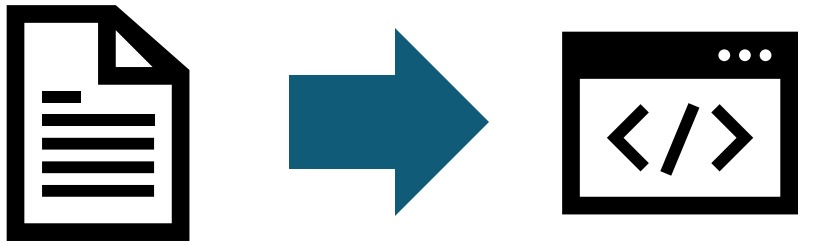
- How do we build an NSR data model?



NSR 80-column
text files

A Data Model for NSR

- How do we build an NSR data model?
  - *Separate data from its original format*

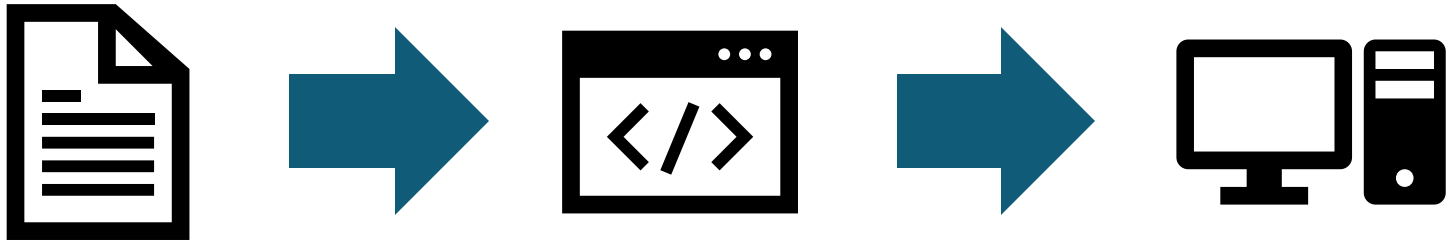


NSR 80-column text files

NSR concepts defined using strings, enums, numbers, etc.

A Data Model for NSR

- How do we build an NSR data model?
  - *Separate data from its original format*
    - Enables translation into other formats (like JSON)



NSR 80-column text files

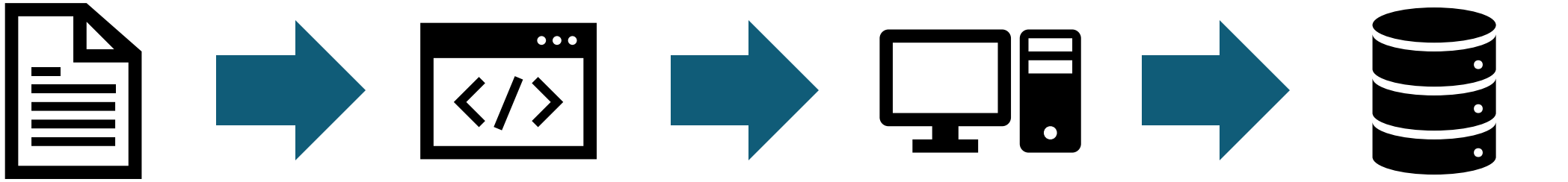
NSR concepts defined using strings, enums, numbers, etc.

Executable code (Java, Python) for reading/writing NSR data

A Data Model for NSR

- How do we build an NSR data model?
  - *Separate data from its original format*
    - Enables translation into other formats (like JSON)
    - Also enables **Object-Relational Mapping (ORM)**

Note: We already have a copy of NSR in JSON!



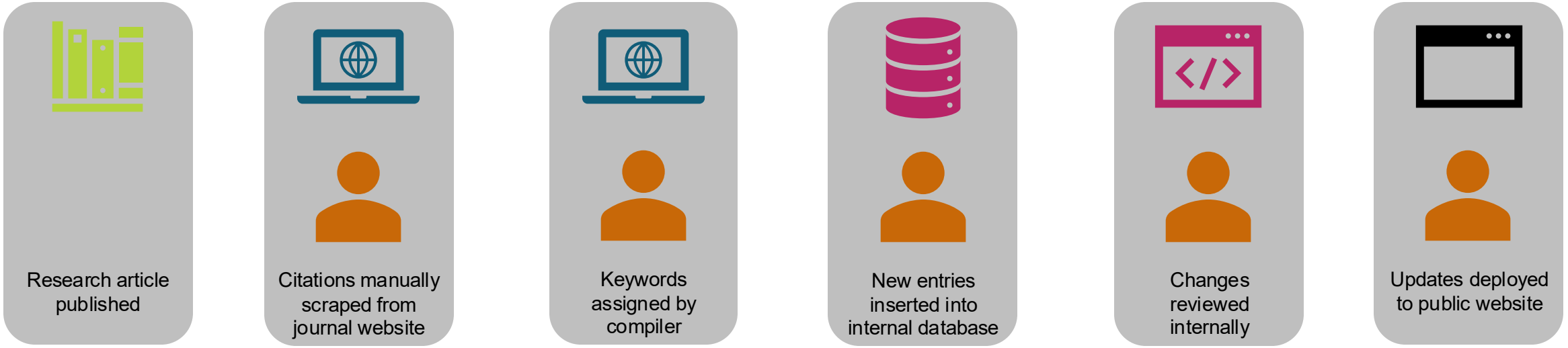
NSR 80-column text files

NSR concepts defined using strings, enums, numbers, etc.

Executable code (Java, Python) for reading/writing NSR data

Ability to recreate NSR database in different formats

Previous Workflow



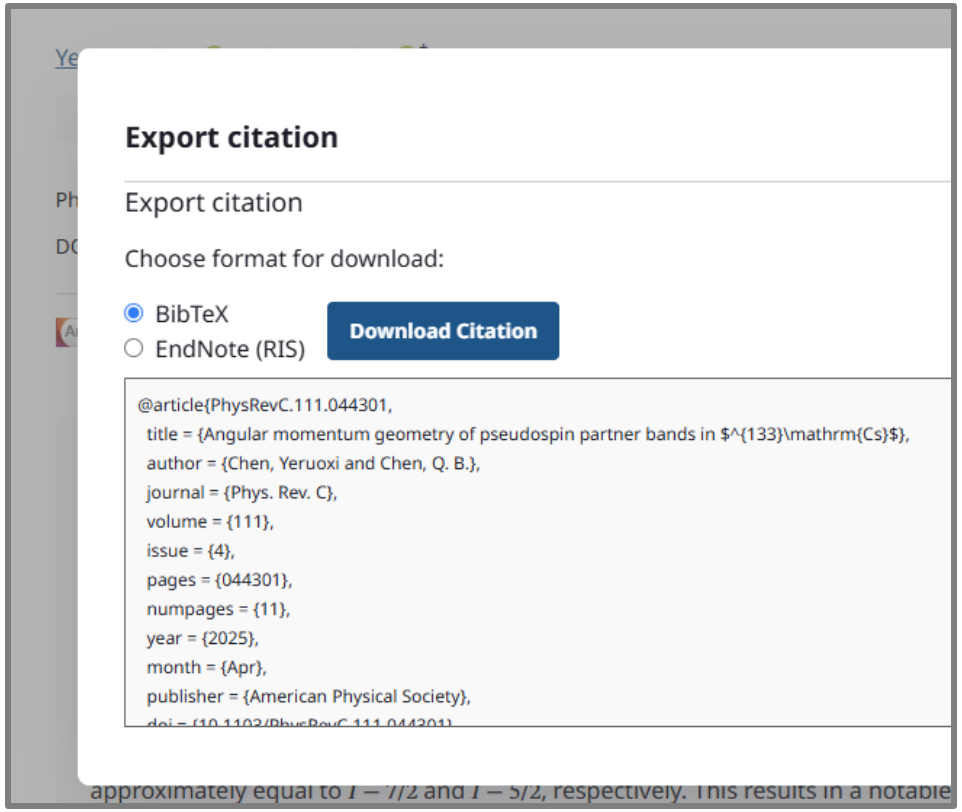
- All steps handled by human beings
  - See previous point about having no contractors

Manual Web Scraping

- Article citations formerly obtained by people searching in websites

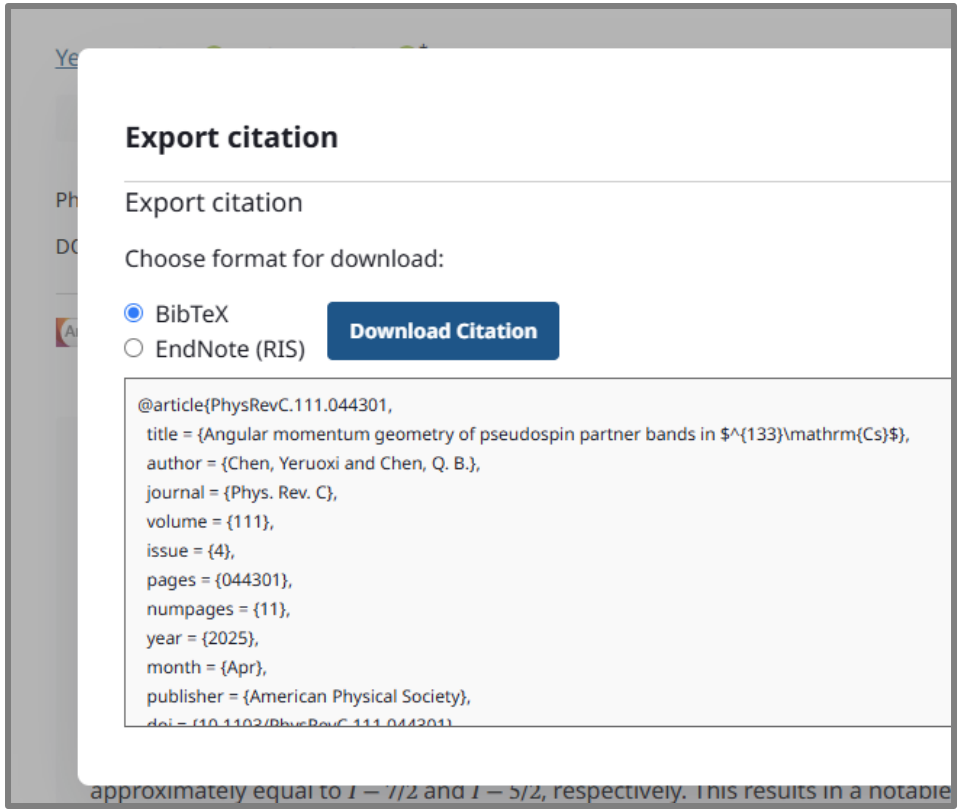
Manual Web Scraping

- Article citations formerly obtained by people searching in websites
  - **Ex. Physical Review C (PRC)**
    - Click on one article, download BibTeX
    - Rinse and repeat for an entire issue
    - Vol 11, Issue 3 – 85 articles added

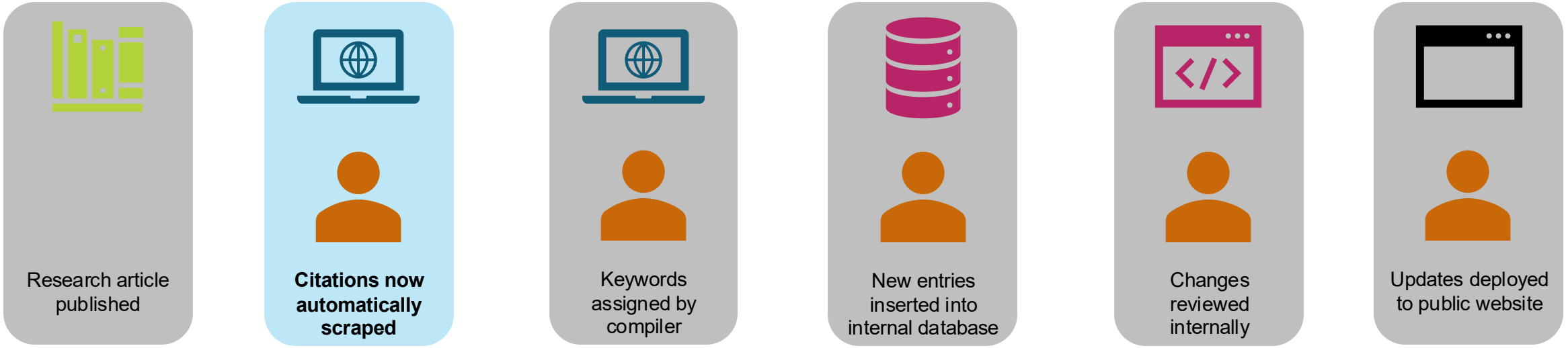


Manual Web Scraping

- Article citations formerly obtained by people searching in websites
  - **Ex. Physical Review C (PRC)**
    - Click on one article, download BibTeX
    - Rinse and repeat for an entire issue
    - Vol 11, Issue 3 – 85 articles added
- **This can be automated**
  (and we already have a scraper!)



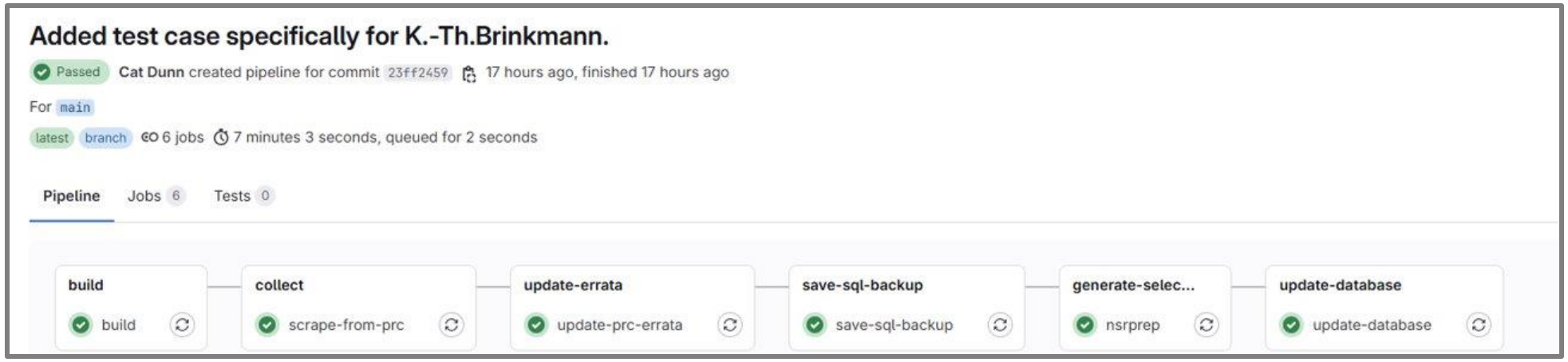
Workflow With Automation



- *Automation saves time/effort that can be used elsewhere*
  - What other steps can we automate?

Proof of Concept Workflow

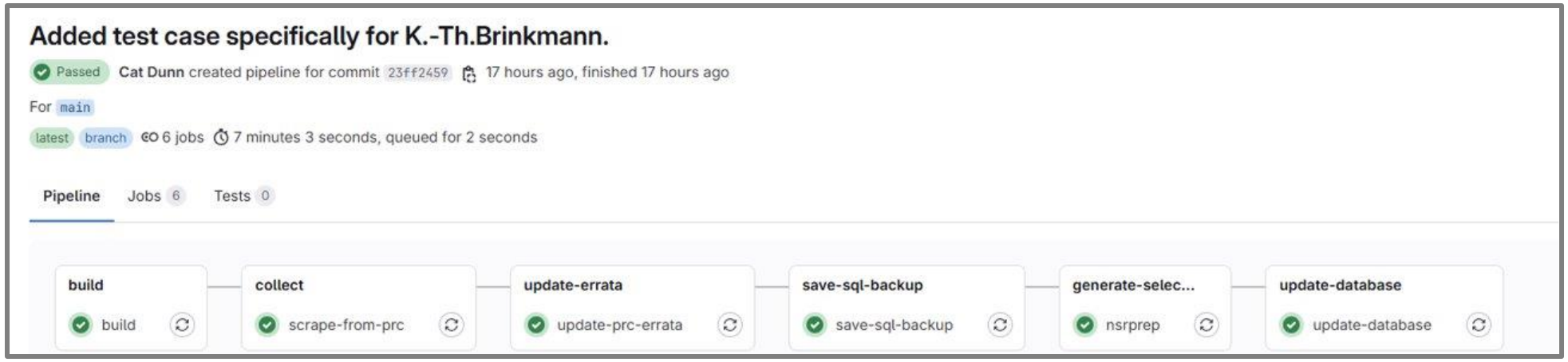
- **update-nsr-sql**: **we have a pipeline!** (based on legacy workflow)



Brookhaven National Laboratory

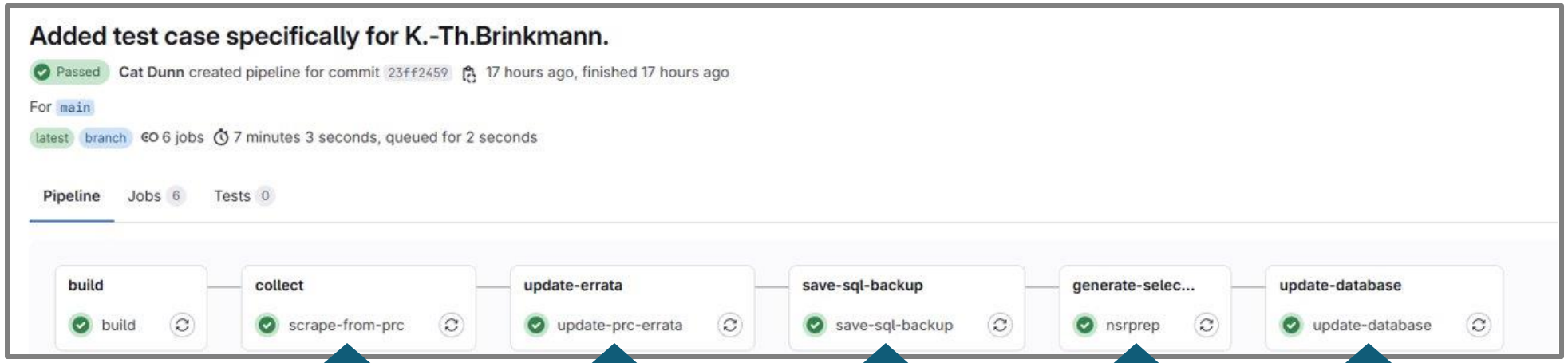
Proof of Concept Workflow

- **update-nsr-sql**: **we have a pipeline!** (based on legacy workflow)
  - Currently only scrapes from Physical Review C



Proof of Concept Workflow

- **update-nsr-sql**: **we have a pipeline!** (based on legacy workflow)
  - Currently only scrapes from Physical Review C



Get latest PRC articles

Update existing NSR entries

Create backup to undo changes if necessary

Generate selectors from keywords

Insert new entries into database

Things to Fix

- Issues with **update-nsr-sql**:

Things to Fix

- Issues with **update-nsr-sql**:
  - Uses old exchange format and previous SQL tables

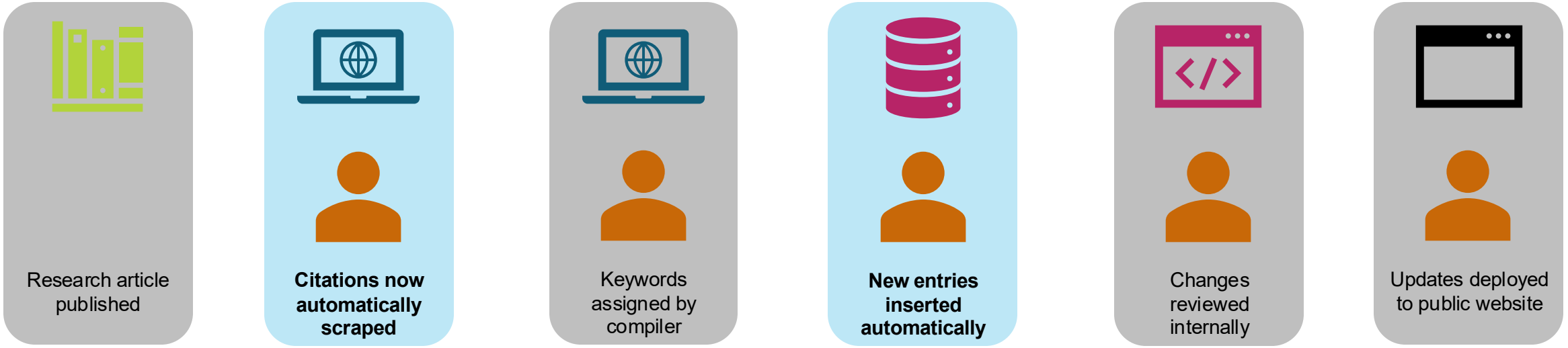
Things to Fix

- Issues with **update-nsr-sql**:
  - Uses old exchange format and previous SQL tables
  - **nsrprep**
    - does not separate format validation and selector generation
    - cannot generate selectors without keywords

Things to Fix

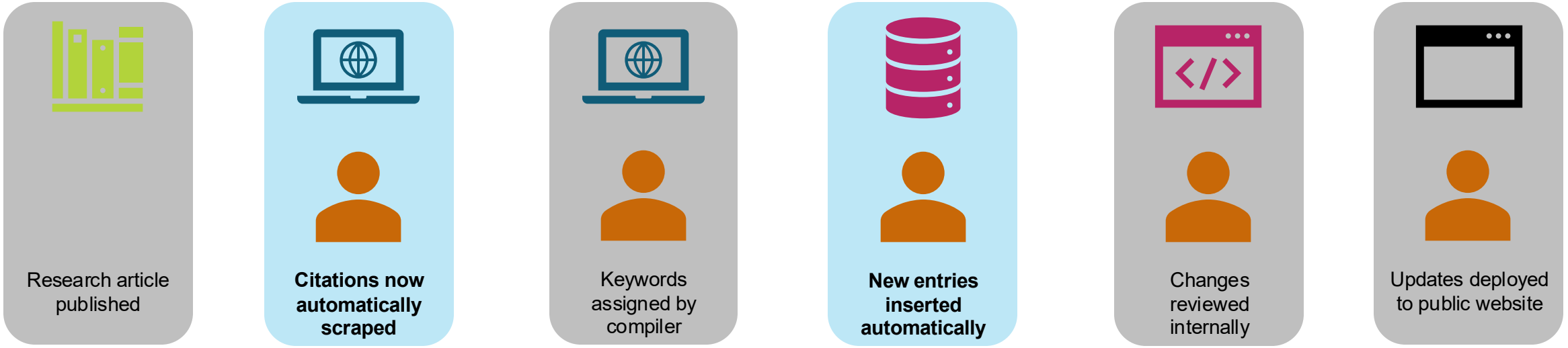
- Issues with **update-nsr-sql**:
  - Uses old exchange format and previous SQL tables
  - **nsrprep**
    - does not separate format validation and selector generation
    - cannot generate selectors without keywords
  - No tools for correcting mistakes
    - typos, name conventions, etc.

Workflow With Automation



- **update-nsr-sql** shows that NSR's workflow can be streamlined
  - We could deploy this framework today, if we wanted to

Workflow With Automation



- **update-nsr-sql** shows that NSR's workflow can be streamlined
  - We could deploy this framework today, if we wanted to
  - See Ramon Arcilla's work on **chatNSR** for automated keywording

What Next?

- How do we prepare NSR for "AI-readiness"?

What Next?

- How do we prepare NSR for "AI-readiness"?
  - ✓ **Create a data model to store existing entries**
    - ✓ **Include the SQL search tables!**
  - ✓ **Translate from exchange format to JSON**
  - ✓ **Automate data collection and insertion**

What Next?

- How do we prepare NSR for "AI-readiness"?
  - ✓ **Create a data model to store existing entries**
    - ✓ **Include the SQL search tables!**
  - ✓ **Translate from exchange format to JSON**
  - ✓ **Automate data collection and insertion**
  - • **Create and enforce a schema to validate data files**
  - • **Build an editor for adding/editing NSR entries**

Schema Validation

- Validator implemented using **Pydantic**

```
2 validation errors for NSREntryReader
publicationYear
  Input should be a valid integer [type=int_type, input_value=None, input_type=NoneType]
    For further information visit https://errors.pydantic.dev/2.11/v/int_type
referenceType
  Input should be 'JOURNAL', 'CONFERENCE', 'REPORT', 'BOOK', 'PRIVATE_COMMUNICATION', 'THESIS' or
'PREPRINT' [type=enum, input_value='', input_type=str]
    For further information visit https://errors.pydantic.dev/2.11/v/enum
```

Schema Validation

- Validator implemented using **Pydantic**
  - Built into Python data model classes
  - Enforces rules for data consistency

```
2 validation errors for NSREntryReader
publicationYear
  Input should be a valid integer [type=int_type, input_value=None, input_type=NoneType]
    For further information visit https://errors.pydantic.dev/2.11/v/int_type
referenceType
  Input should be 'JOURNAL', 'CONFERENCE', 'REPORT', 'BOOK', 'PRIVATE_COMMUNICATION', 'THESIS' or
'PREPRINT' [type=enum, input_value='', input_type=str]
    For further information visit https://errors.pydantic.dev/2.11/v/enum
```

Schema Validation

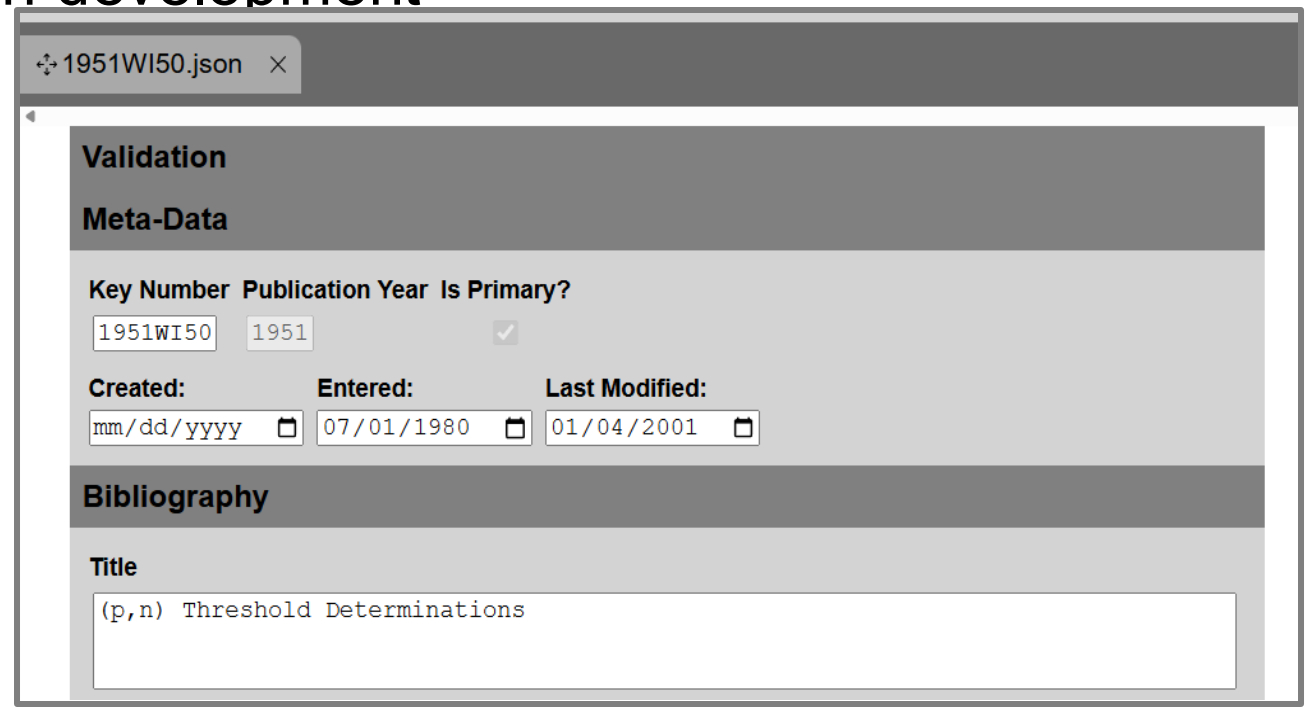
- Validator implemented using **Pydantic**
  - Built into Python data model classes
  - Enforces rules for data consistency

```
2 validation errors for NSREntryReader
publicationYear
  Input should be a valid integer [type=int_type, input_value=None, input_type=NoneType]
    For further information visit https://errors.pydantic.dev/2.11/v/int_type
referenceType
  Input should be 'JOURNAL', 'CONFERENCE', 'REPORT', 'BOOK', 'PRIVATE_COMMUNICATION', 'THESIS' or
'PREPRINT' [type=enum, input_value='', input_type=str]
    For further information visit https://errors.pydantic.dev/2.11/v/enum
```

  - *Unlike **nsrprep**, this can run offline without connecting to NSR*

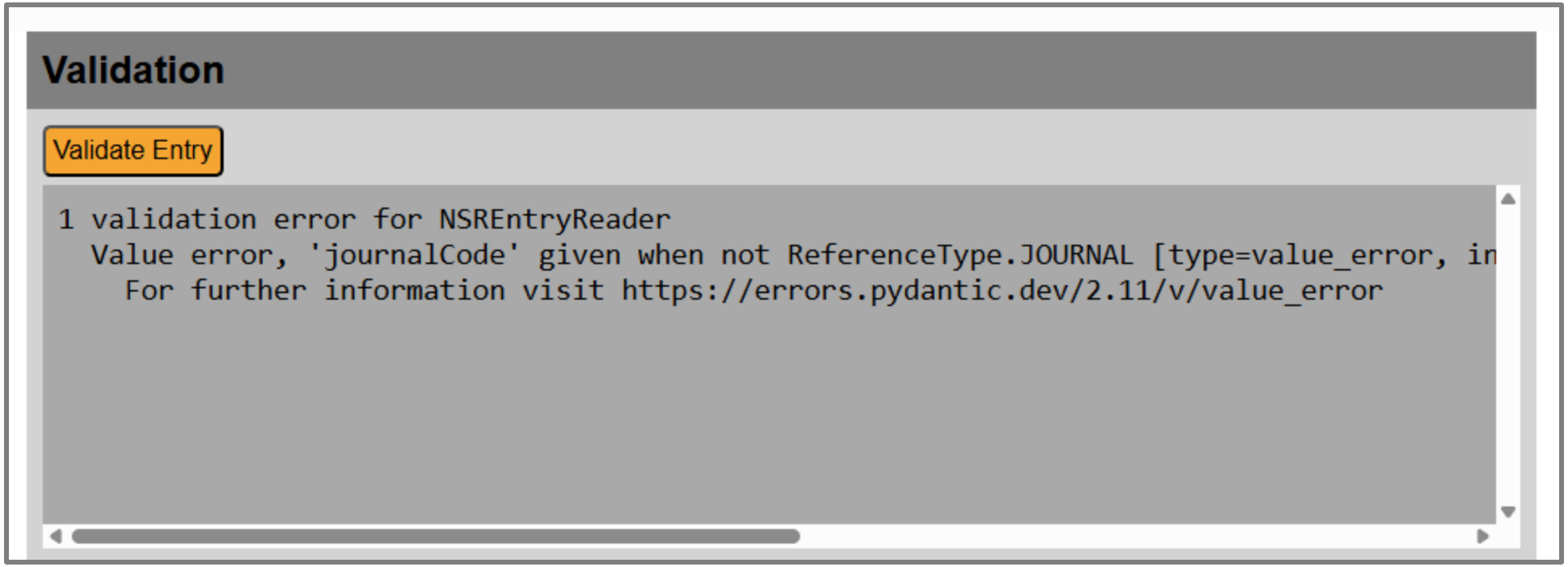
NSR Editor

- An application for creating/editing NSR JSON files
  - Currently in development



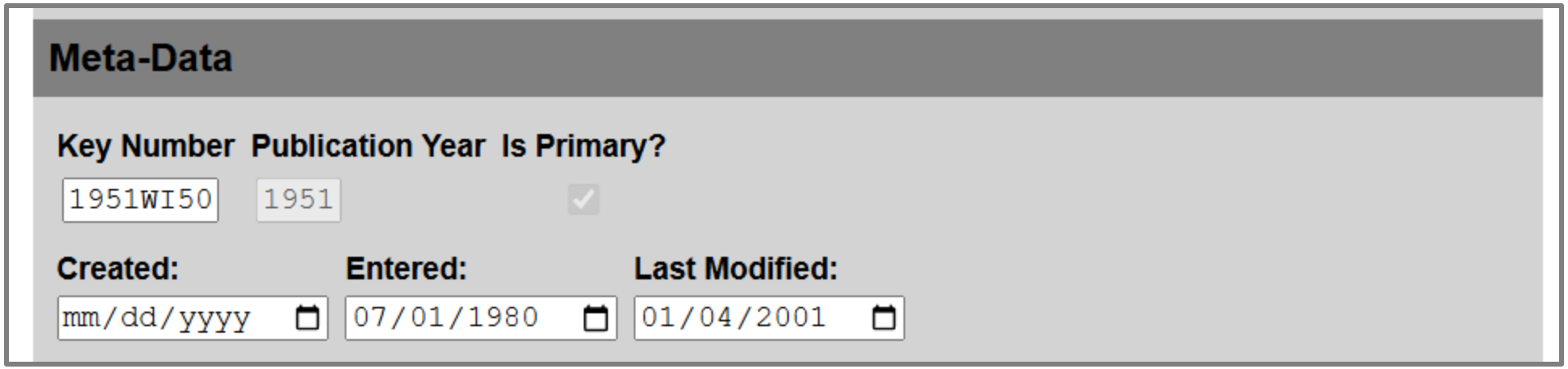
NSR Editor (contd.)

- Built-in validator to catch format errors



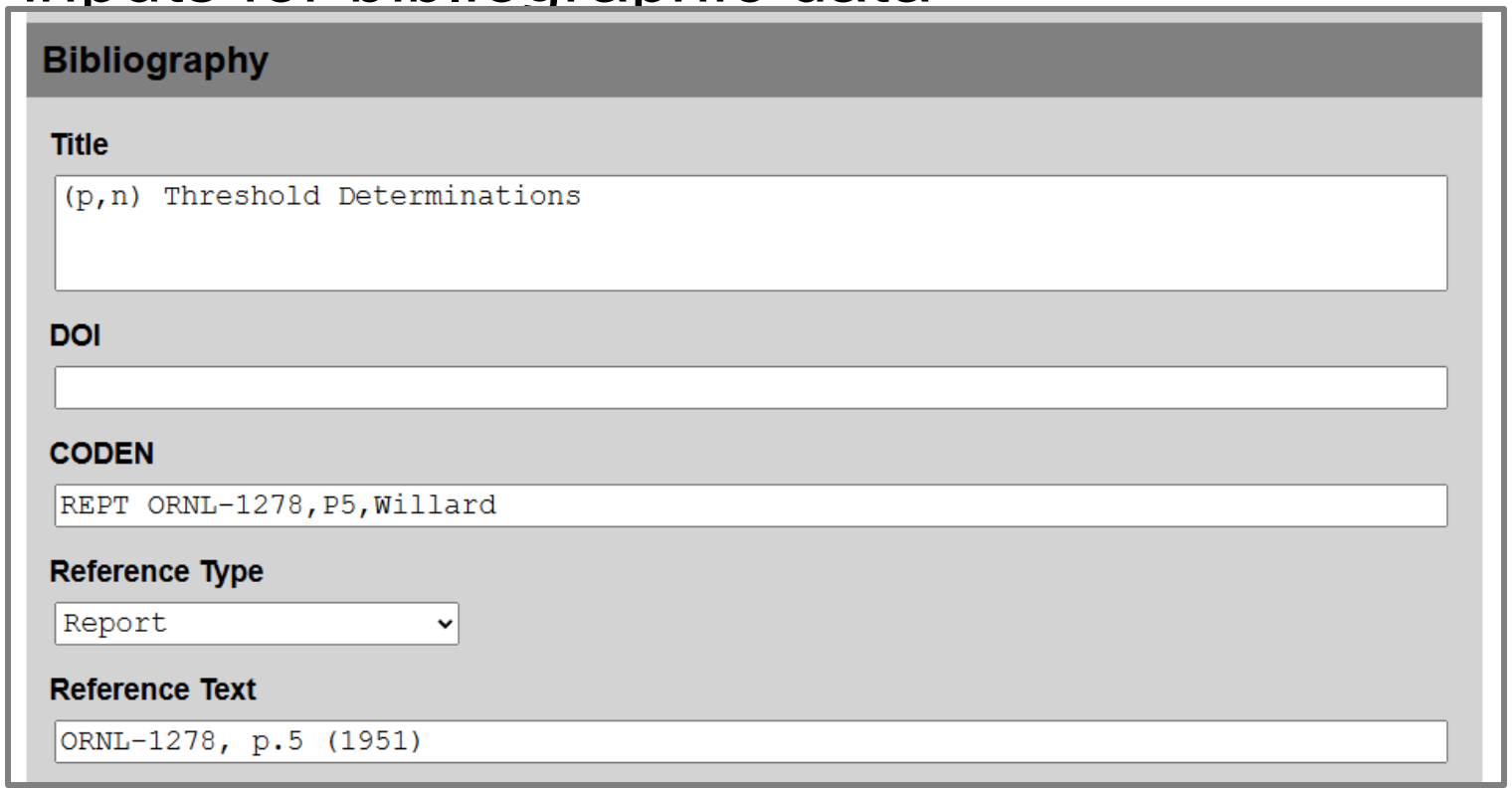
NSR Editor (contd.)

- Has inputs for identifying information



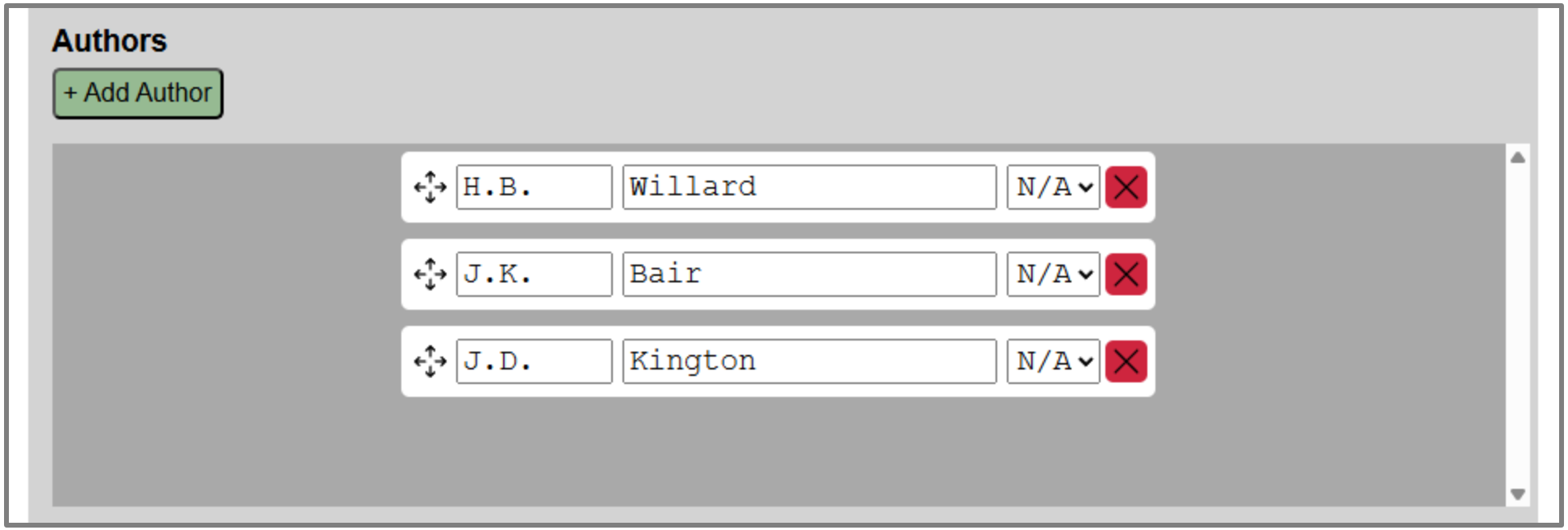
NSR Editor (contd.)

- Also has inputs for bibliographic data



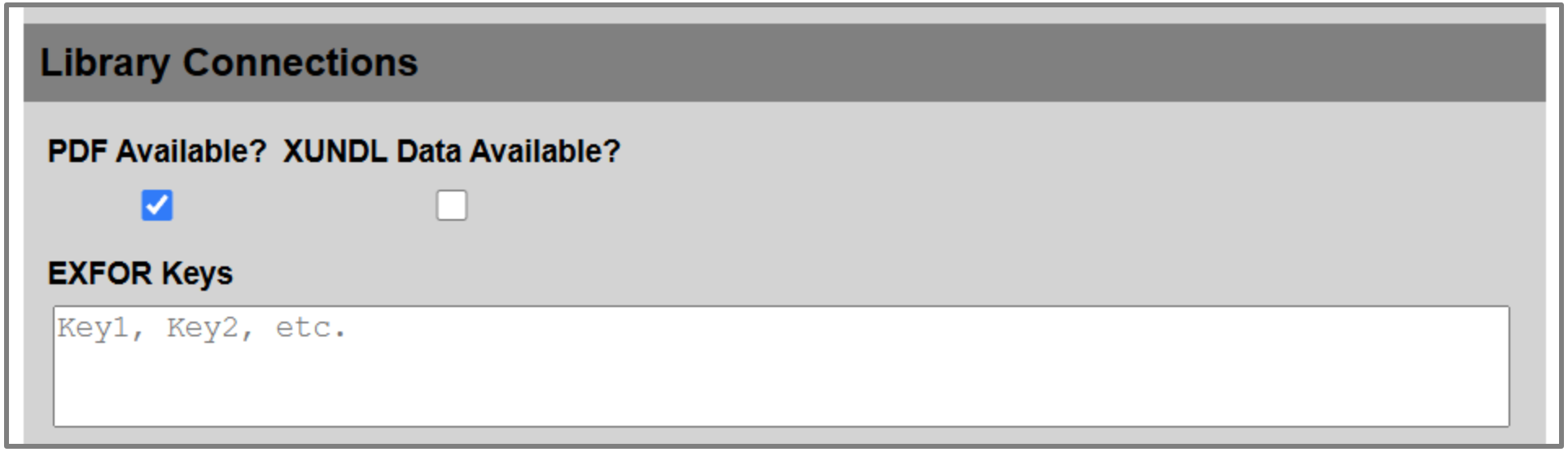
NSR Editor (contd.)

- Enables adding and re-ordering authors/collaborations



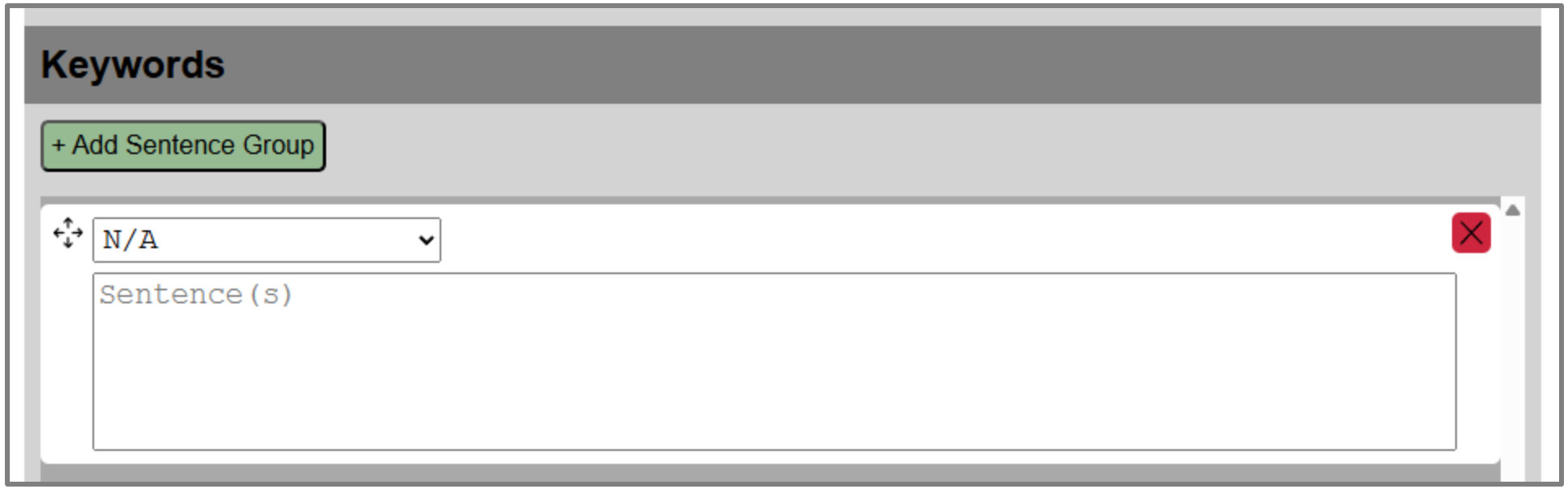
NSR Editor (contd.)

- Enables adding links to supporting datasets



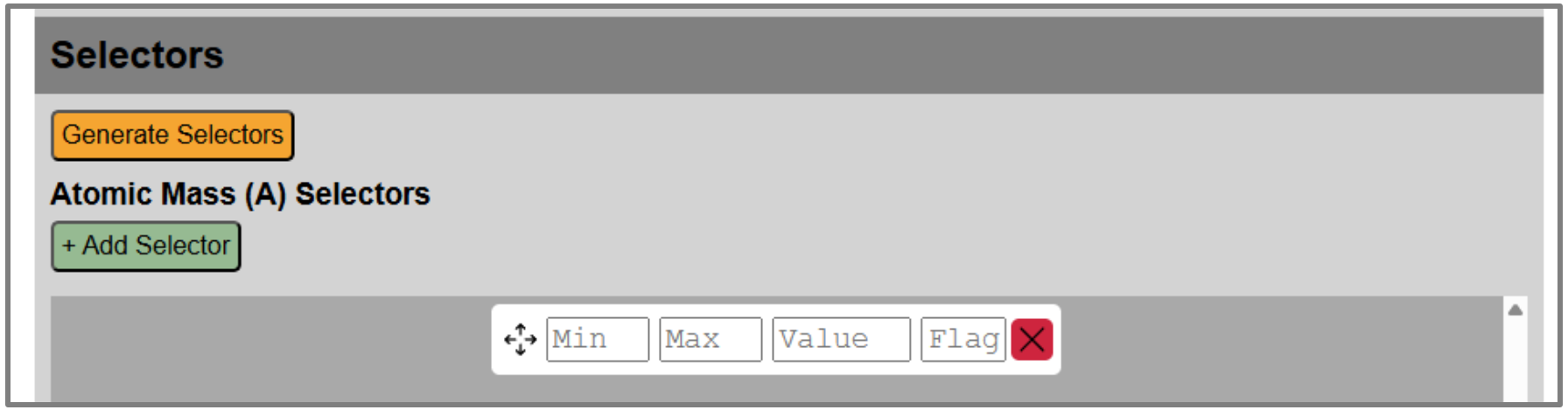
NSR Editor (contd.)

- Includes text inputs for adding keywords



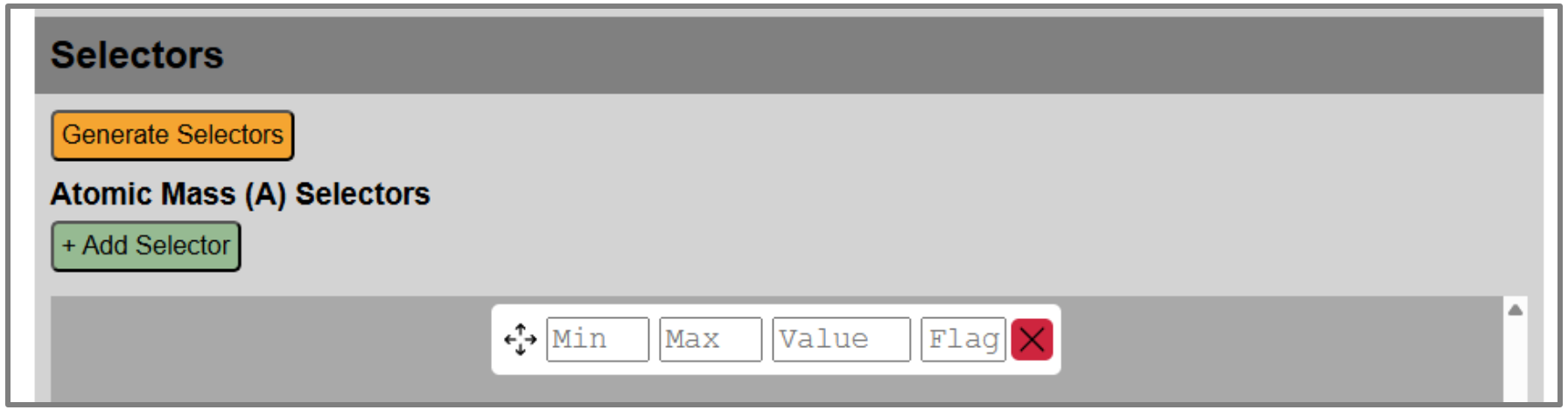
NSR Editor (contd.)

- Also includes space for manually adding selectors



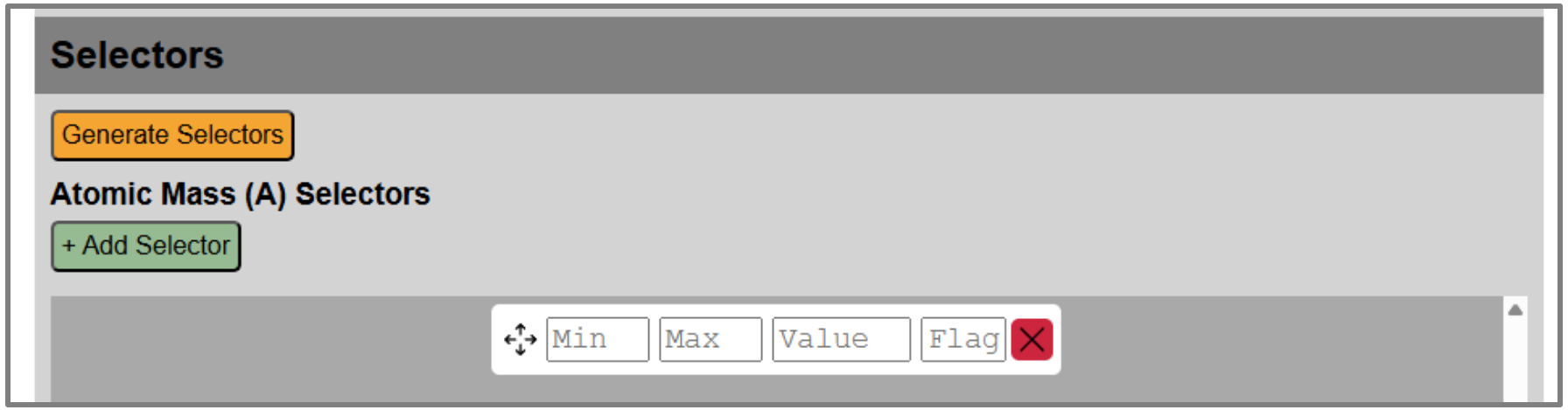
NSR Editor (contd.)

- Also includes space for manually adding selectors
  - **Reminder:** *these power NSR's search functions!*



NSR Editor (contd.)

- Also includes space for manually adding selectors
  - **Reminder:** *these power NSR's search functions!*
  - In development: automatic selector generation from keywords



Where To From Here?

- Full JSON workflow not yet complete

Where To From Here?

- Full JSON workflow not yet complete
  - Need to scrape from more journals
    - Nuclear Physics A
    - European Physics Journal

Where To From Here?

- Full JSON workflow not yet complete
  - Need to scrape from more journals
    - Nuclear Physics A
    - European Physics Journal
  - Need more keyword assignment
    - (see Ramon's presentation on **chatNSR**)

Where To From Here?

- Full JSON workflow not yet complete
  - Need to scrape from more journals
    - Nuclear Physics A
    - European Physics Journal
  - Need more keyword assignment
    - (see Ramon's presentation on **chatNSR**)
  - NSR editor needs further testing

Where To From Here?

- End goal: transition to full JSON database
  - *NSR exchange format has not been replaced*

Where To From Here?

- End goal: transition to full JSON database
  - *NSR exchange format has not been replaced*
  - Changes will be made to **nsr-update** first:
    - https://www.nndc.bnl.gov/nsr-update/

Where To From Here?

- End goal: transition to full JSON database
  - *NSR exchange format has not been replaced*
  - Changes will be made to **nsr-update** first:
    - https://www.nndc.bnl.gov/nsr-update/
  - Already completed:
    - Re-building SQL database from JSON using ORM
    - Searching for entries from ORM database
    - Displaying LaTeX titles/keywords on web