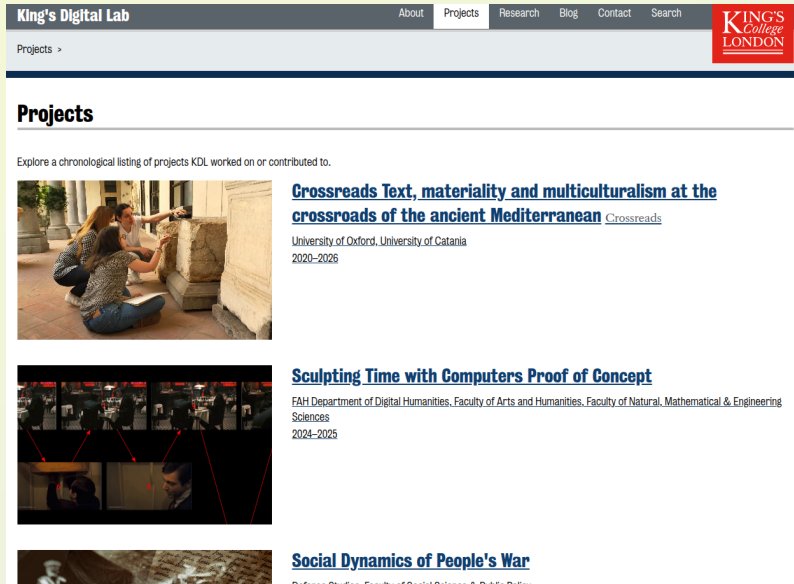


Static First Development Approach

King's Digital Lab, King's College London

Adapted from a presentation by Miguel Vieira

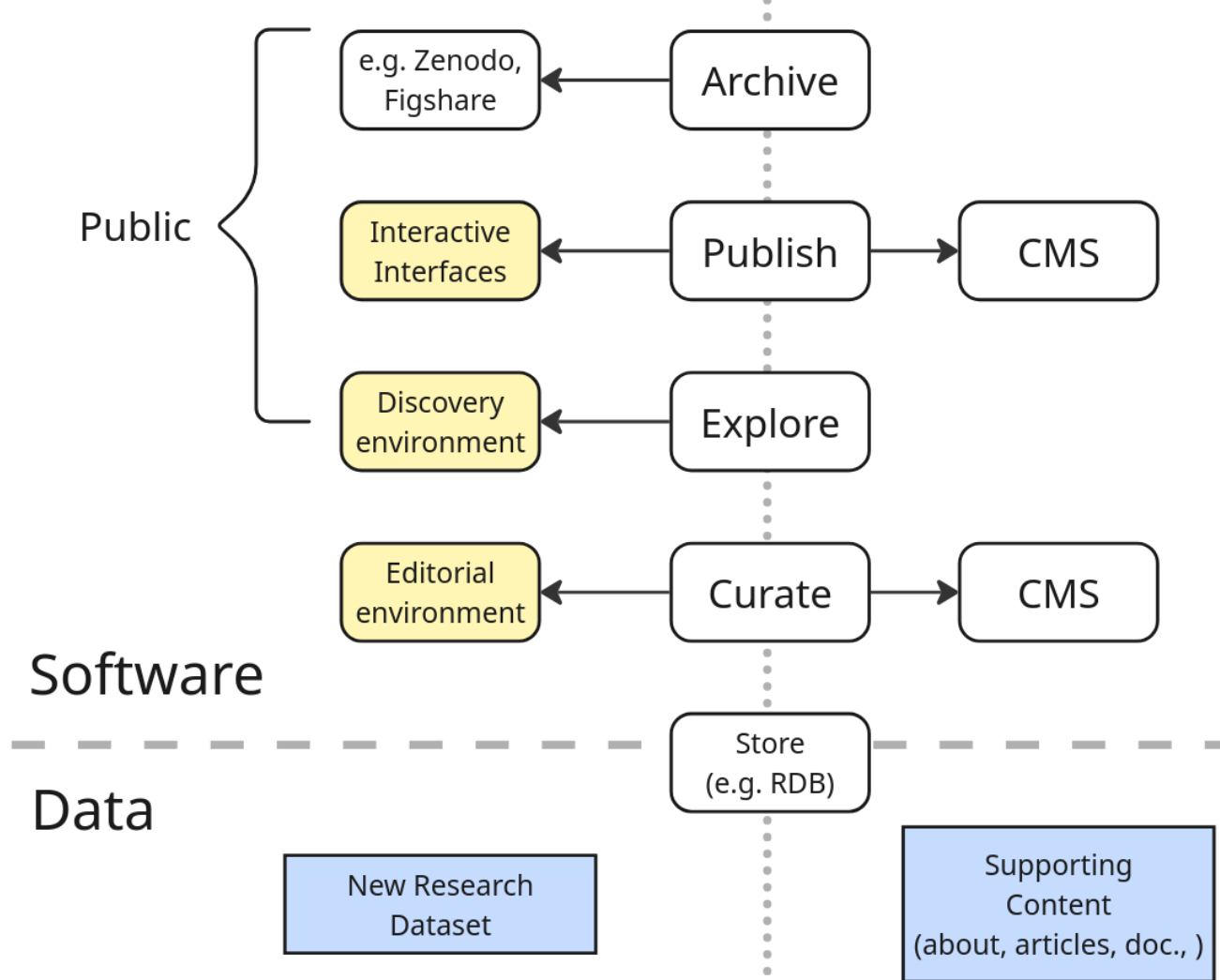
1. Some background



One main service offered by King's Digital Lab is to **develop, host and maintain** advanced **digital resources** to support core objectives of funded research projects in the Arts & Humanities.

kdl.kcl.ac.uk/projects/

Digital Resource



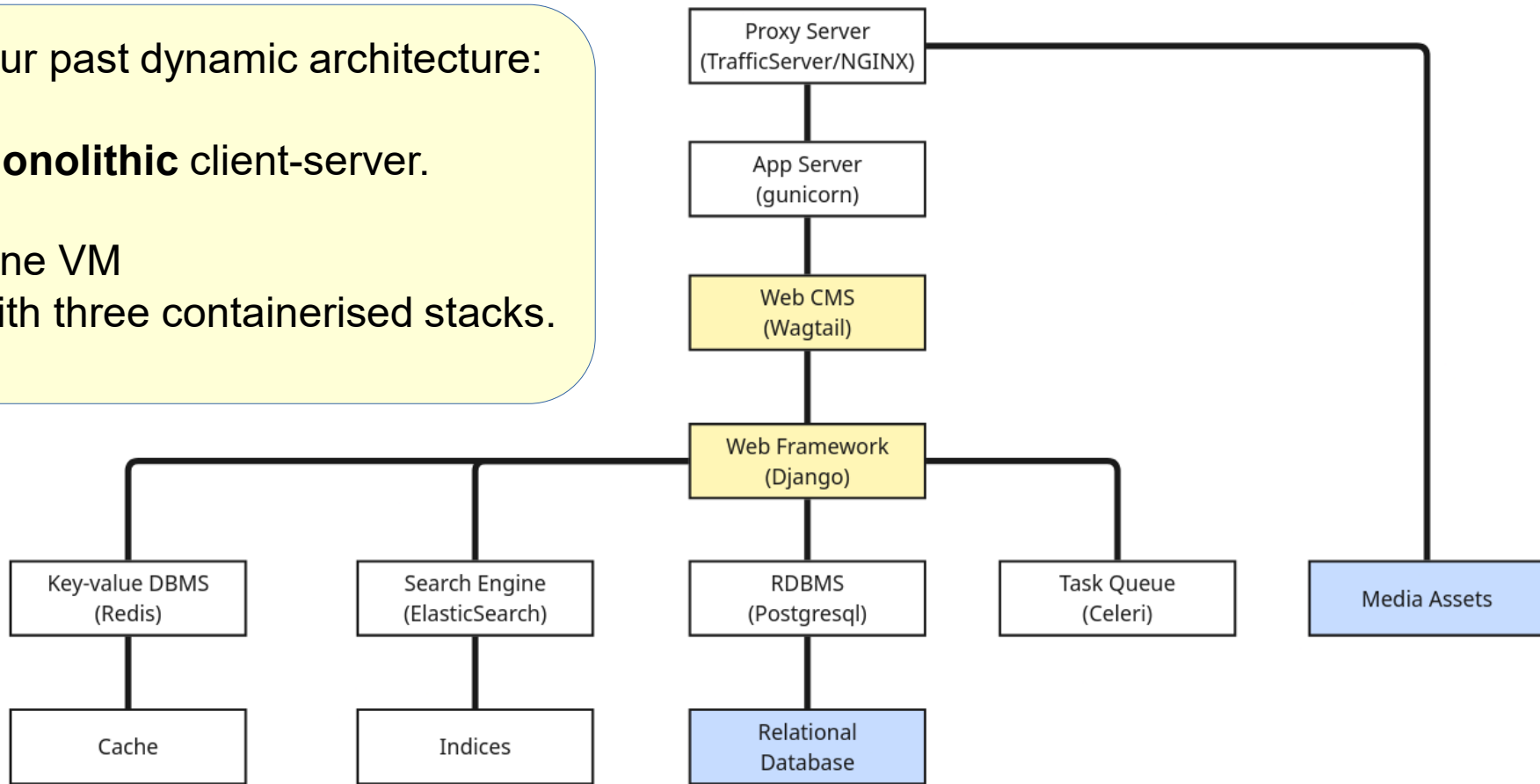
Life cycle of a resource

	Agile development (active project period, 1-5 years)	Maintenance (feature freeze , 1-10 years)	Static Last (data freeze)	Public archival
Publication	Early web presence w/ regular releases	Break fix support only	Static snapshot of front-end site	
Curation and exploration	Build internal components	Break fix support only	Turned off and archived	All code on github from the project start
Data	Build data store & management	Break fix support only (still editable)	Disconnect & archive Database	Deposition to e.g. Figshare
Open access	Open access	Open access	Open access	

Our past dynamic architecture:

Monolithic client-server.

One VM
with three containerised stacks.



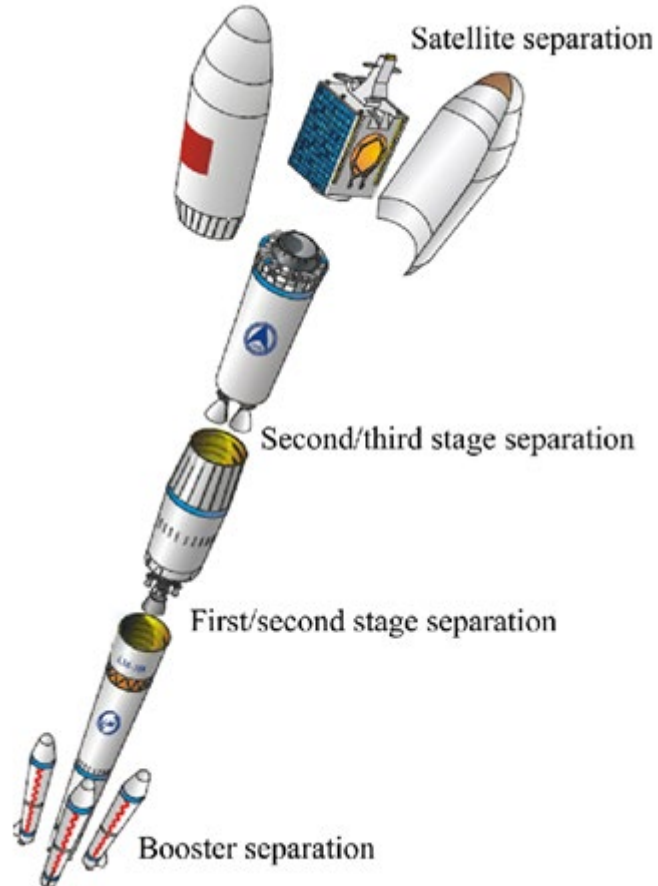
Static Snapshot

1. Snapshot with wget or Sitesucker
2. Re-implement **any server-side rendering that is dependent on parameters provided by a user** (e.g. Ajax requests; Search page)
3. Challenging if **server-side processing too complex or the pool of input data is too vast** to be shifted to the browser
4. Involves feasibility assessment, negotiation/**compromises** with researchers and loss or simplification of features
5. Visual regression **testing** and review by other member of the team

Practical observations

- Static snapshot of decade(s) old **legacy** resources can take many days for a developer unfamiliar with the features and code
 - Hard to catch all functional regressions after maintenance or snapshot
- **Small Data**: many simpler resources didn't even need such a full dynamic stack in the first place
- Client devices (RAM, CPU, Network), browsers and frameworks are much more **capable** than 10-15 years ago
- RDBs are not the most **archive**-friendly format. More of a data dump than an enabler for future research

2. Static First approach



What is **Jamstack**?

Jamstack is an architectural approach that decouples the web experience layer from data and business logic, improving flexibility, scalability, performance, and maintainability.

Jamstack removes the need for business logic to dictate the web experience.

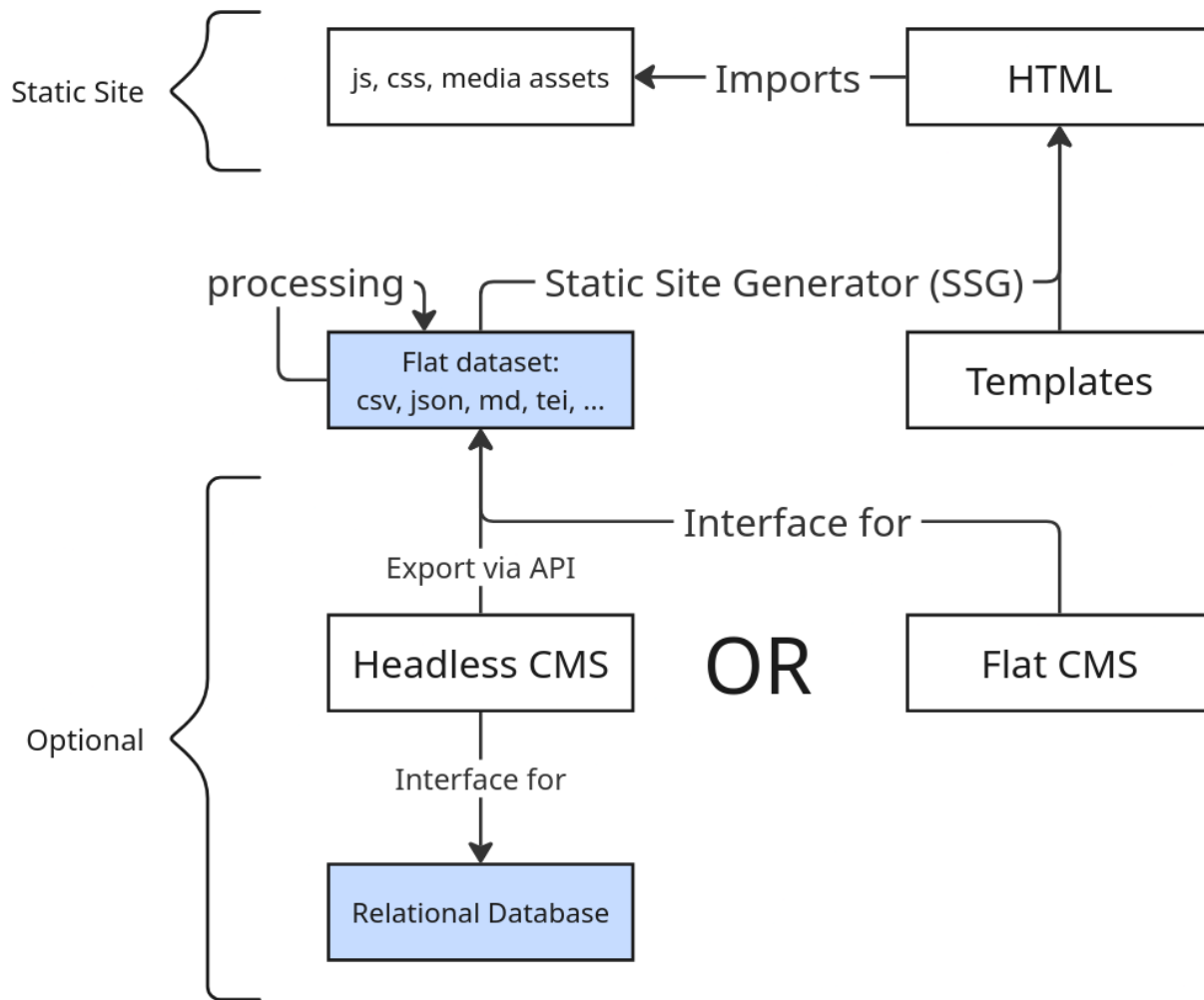
It enables a composable architecture for the web where custom logic and 3rd party services are consumed through APIs.

jamstack.org

Cen, Z., Gao, R. F., and Shen, L., LM-3A Series Launch Vehicle User's Manual, China Academy of Launch Vehicle Technology, Beijing, 2011, pp. 13–14, 2–10, Chaps. 2, 5

Key ingredients

1. A **minimal** (server-side) stack by default
2. made of **interchangeable, disposable** and **decoupled** components
3. using **files** as “API”
4. in a machine & human **readable format**
5. and **schema** compliant (default to schema.org)



Benefits

1. Less demanding: low maintenance, low resource
2. Highly sustainable and portable
3. More interoperable
4. More secure, fast and robust
5. Better code reuse b/c JS on client & server sides

jamstack.org/why-jamstack/

Component 1: Static Site Generator

- When? our new default for most projects
- Options: 11ty, SvelteKit, Jekyll, Next.js
- Example: KDL's own website

Component 2: Flat File CMS (optional)

- When? Editors prefer a form to edit metadata and content in flat files (e.g., Markdown). Suitable for smaller, content-driven projects.
- Options: Front Matter, Decap CMS

The image shows a side-by-side comparison of a CMS editor interface and its output. On the left is a form for editing a blog post, and on the right is the corresponding Markdown frontmatter code.

Form Interface (Left):

- Title:** King's College London joins DARIAH as Cooperating Partner
- Description:** (Empty text area)
- Publishing date:** 11/06/2023 00:00 (with a 'now' button)
- Content preview:** A dashed box containing an image icon and the text 'Add your content preview'.
- Is in draft:** A toggle switch that is currently turned on.

Markdown Frontmatter (Right):

```
frontend > src > blog > kcl-joins-dariah.md >
1 ---
2 title: King's College London
3 subtitle: First Published on
4 tags:
5   - post
6 authors:
7   - Alessandra Esposito
8 date: 2023-11-06
9 excerpt: "The Digital Research
10 announce it has renewed a Co
11 Kingdom. "
12 feature:
13   image: /assets/images/blog,
14   title: Photo by Diva Pla
15   description: Close up photo
16 draft: false
17 ---
18 The Digital Research Infrastru
19 renewed a Cooperating Partner
20 United Kingdom.
21 "The enthusiastic about th
```

Component 3: Headless CMS (optional)

- When? for projects that require advanced content modeling (more interrelated entities with richer structure)
- Options: Directus, Pocketbase
- Example: KDL's own website

3. Where are we now?

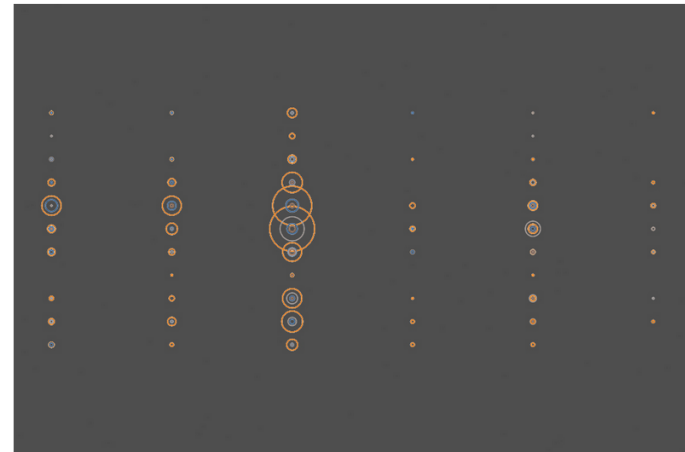
Most of our resources are now Static First. Experience is very positive. But we still need to:

- harmonise our default folder structure across projects
- improve data modeling practice
(ORMs used to make us more disciplined)
- settle on default SSG framework
- build more reusable patterns & components

4. Examples

1. KDL's own website
2. CROSSREADS: an environment to annotate letterforms from thousands of ancient inscriptions found in Sicily – Jonathan Prag (PI), Simona Stoyanova (RA)

Living with Machines LwM



A multidisciplinary research collaboration between the British Library and the Alan Turing Institute with King's College London, the Universities of Cambridge, East Anglia, Exeter, and London (QMUL), the [Living with Machines](#) project aimed to investigate how the introduction of machines into the workplace and landscape changed the lives of ordinary people in the nineteenth century. It had several research strands.

github.com/kingsdigitallab/kdl

Project Status: Post-project

Project links

- [Repository](#)
- [Interactive notebook: Language of Accidents](#)
- [Interactive notebook: Language of Mechanisation](#)
- [UKRI URL](#)
- [Language of Mechanisation data paper](#)

KDL KDL

Agent Organisation Person Definedtermset Definedterm Project Roles Linkrole Role Webpage

Project Living with Machines

CMS Status

Published

Name *

Living with Machines

Project title

Slug *

[living-with-machines](#)

Creative Work Status

Active

Project status

Alternate Name

LwM

Acronym, nickname

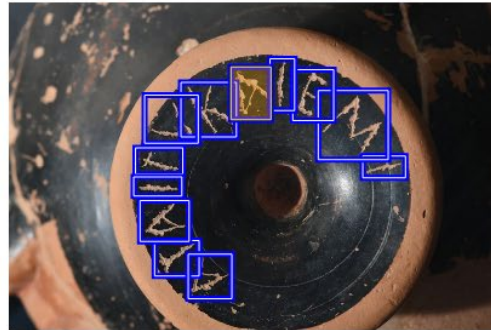
Site generated by 11ty from flat files snapshot of the CMS content

- Headless CMS: Directus
- RDBMS: Postgresql

Object [↗](#)

ISic020319

Image

ISic020319.jpg [▼](#)Image [↗](#) - 12 annotations - Types☐ Hide boxes[🕒](#) Read only (Log in)Transcription [↗](#)☐ Show supplied text

1

A T A I T Y K A I ε μ ι

Description [↗](#) (Selected)

Script

Greek [▼](#)

Allograph (A-greek)

A [▼](#)

Tags

[Add](#)

type1.1.1

Components

apex

☒ angular☐ rounded☒ sans serif☐ serif☐ square

crossbar

☐ absent☒ ascending☐ broken☐ circular☐ curved

kingsdigitallab.github.io/crossreads

github.com/kingsdigitallab/crossreads

Crossreads - Specifics

- Image server is the only external service
- No SSG, just **pure html, js (vuejs) and css**
- **Curation**, discovery and publication of research data is done on the public site hosted as github Pages (i.e. dual use interface)
- **Bulk edit** runs asynchronously with github workflows
- Two **modes**: write (with a Github token, for the RA) and read-only (without, i.e. public visitor)
- Used to create 5900+ annotations
- All data versioned in **standard formats** (TEI, W3C web annotations, DTS) on github