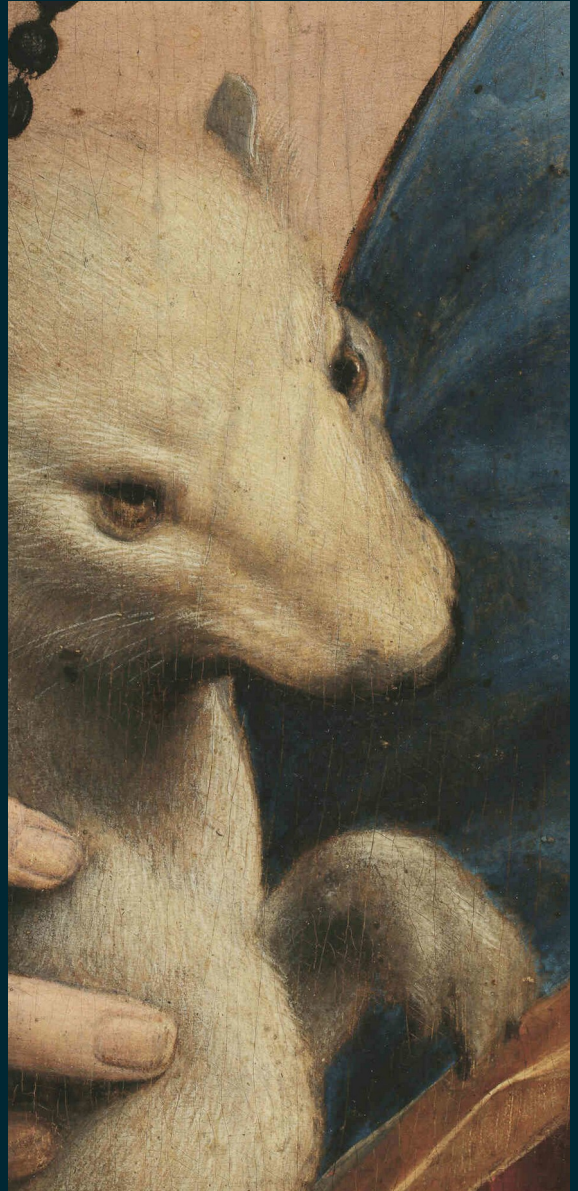
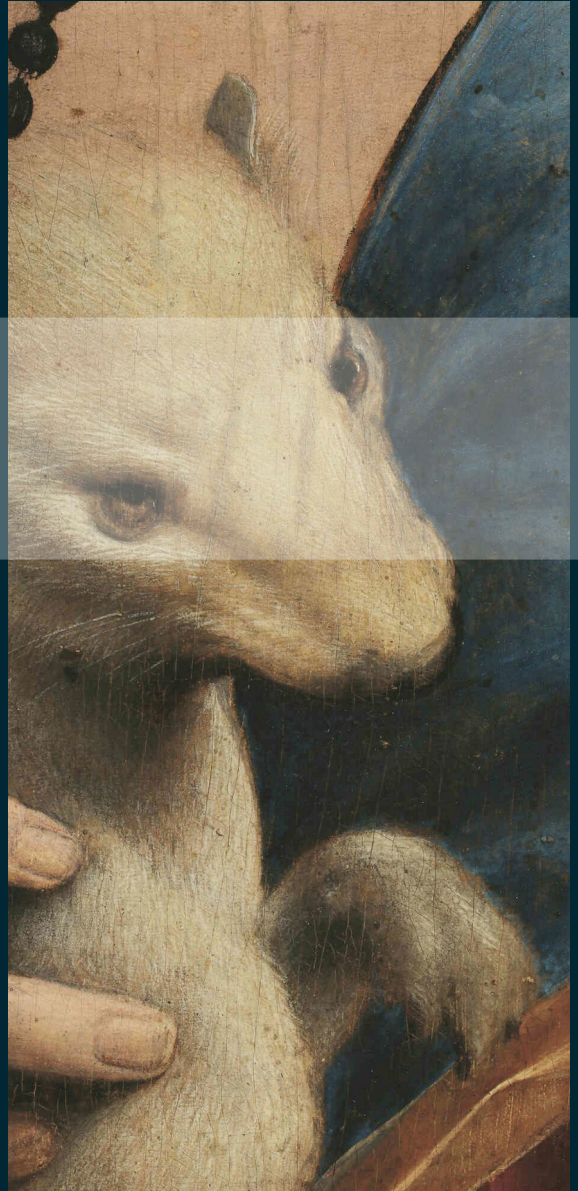
The Hermine Project

Open source code and data for legal compliance



Ospo OnRamp
September 16th, 2022

Who we are



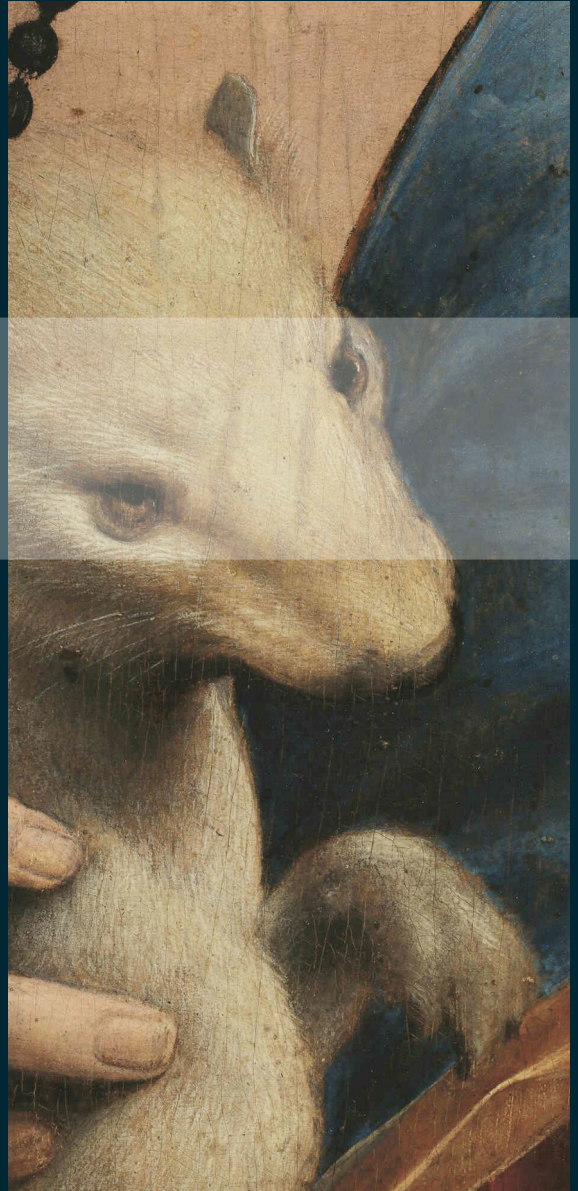
A FOSS, community driven project

The Hermine project has been started last year by six, end-user, partner companies in a semi-formal context.

- no dedicated organisation, yet

- 3 committees (legal, technical, steering)

Where we are

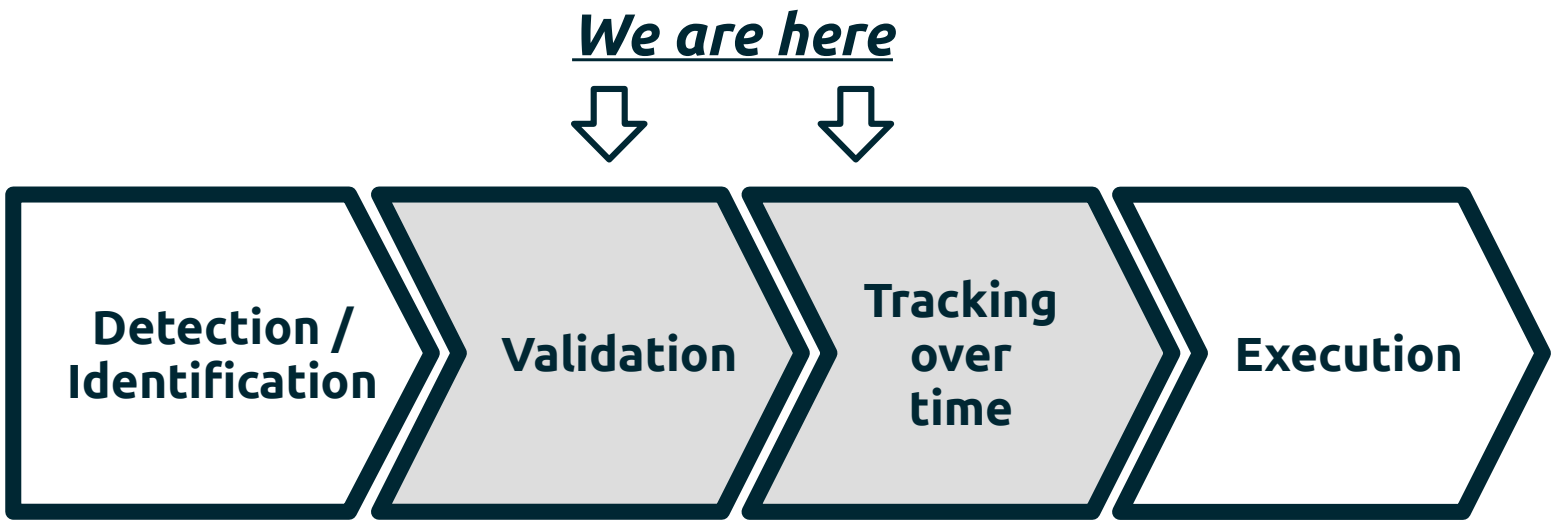


Position in the FOSS compliance landscape

We rely heavily on (and try to contribute to) existing tools and standards:

- OSS Review Toolkit
- SPDX
- NexB's Scancode Toolkit
- FOSSology
- PURL
- Etc.

Position in the FOSS compliance landscape

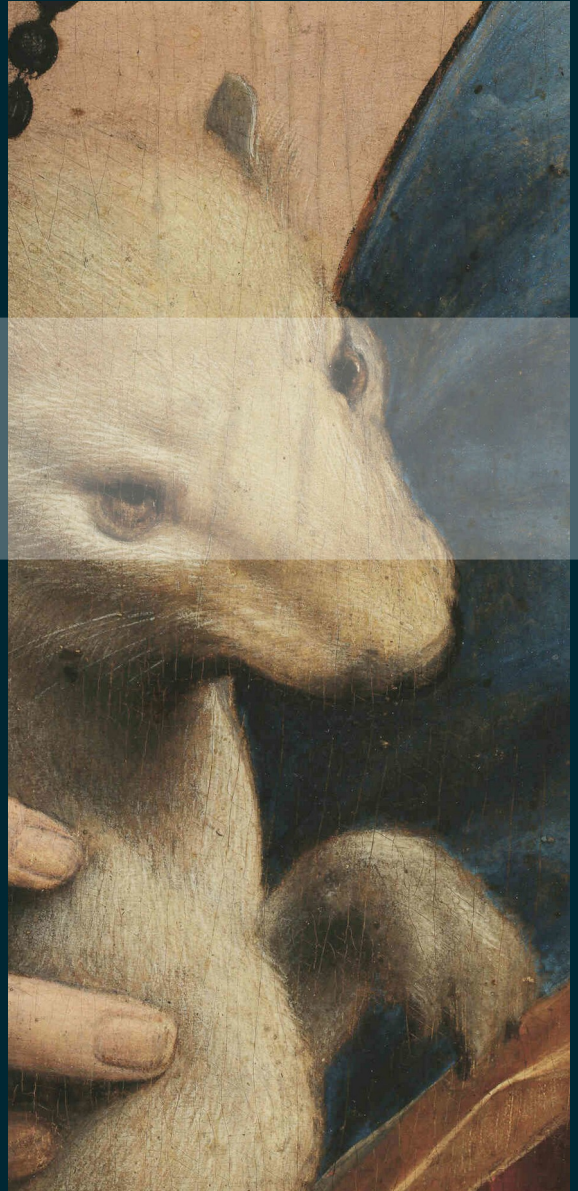


Other SBOMs related topics

We currently focus on legal aspects (license compliance, export control), but we aim to take into account at later stages of development:

- Sustainability (Dependency funding)
- Security

What we do (⚖)



Foreword on the general spirit towards compliance

Because the Hermine tool is designed by end users, it’s goal is to be efficient and pragmatic while significantly limiting legal uncertainty.

We want each organisation to be able to decide on the level of risk they consider acceptable.

We analyse licences

One goal of the project is to provide a systemic framework to analyse FOSS (or nearly-FOSS) licences, so that:

- They can be handled programmatically
- Legal departements can share their interpretations
- It’s easier to reach a concensus about interpretations, hence increasing legal predictability

We analyse licences : global characterics

For each licence we a set of characteristics, like:

- The copyleft level
- The nature of rights granted (is there a patent grant, a restriction for commercial usage,…)
- The choice of law & venue
- etc.

This set is still being worked on by the legal committee

EPL-1.0

Identity

Spdx id:	EPL-1.0
Long name:	Eclipse Public License 1.0
Url:	**Currently:** http://www.eclipse.org/legal/epl-v10.html **Change:** http://www.eclipse.org/legal/epl-v10.html
Osi approved:	Yes
Fsf approved:	Yes
Law choice:	laws of the State of New York and the intellec
Venue choice:	

Evaluation

Status:	Checked The review status of the license
FOSS Policy	Allowed depending on context
FOSS Policy explanation:	

We analyse licences : obligations

We breakdown every licence in a set of obligations, mentioning for each how it is triggered:

- If it has been modified or not ;
- For which type of exploitation (e.g. : source distribution, network interaction, or plain usage – for passive obligations)

If the meaning of the obligation is very common (e.g. there is more than 200 ways to say “copy the licence in the documentation), we link it to a *generic obligation*.

This way, you care only about creating a process to implement the generic obligation, not the 200 individual ones.

OBLIGATIONS

Obligation: BSD-3-Clause -Full License in documentation

Generic:	License and copyright notices in documentation
Name:	Full License in documentation
Verbatim:	2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
Passivity:	Active
Trigger expl:	If the component is distributed as binary
Trigger mdf:	Whether the component is modified or not

We create Open Source license policies

- For each licence, you can define if it’s acceptable by your organisation
- This acceptability can be linked to some technical criterion (e.g. allowed for dynamic linking, not static linking) or business context (e.g. in a product pertaining to certain categories only - WIP)

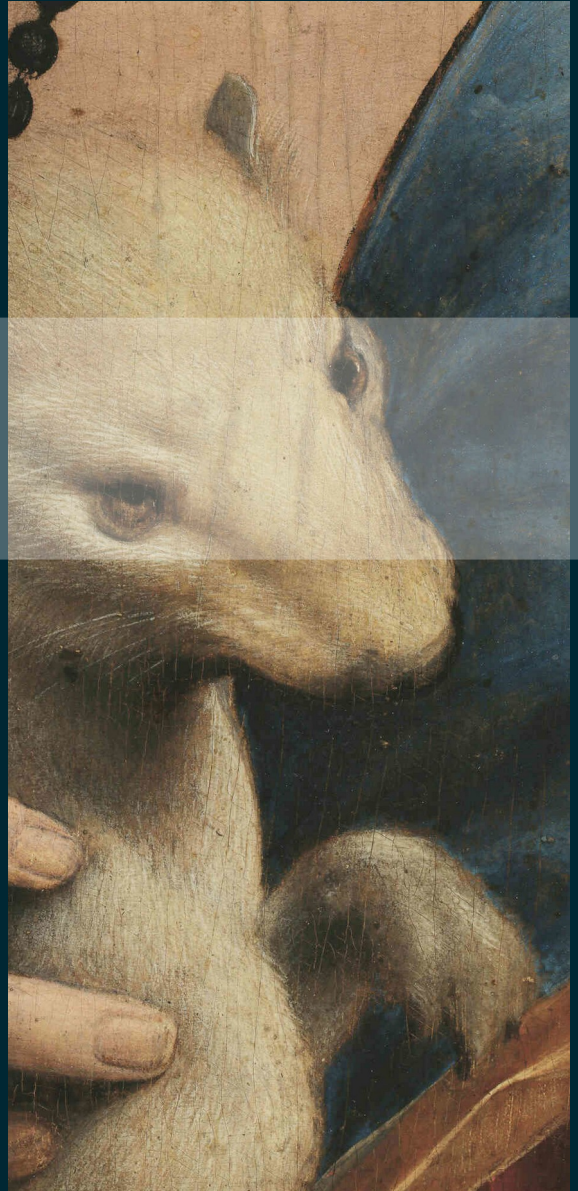
460 Licences

Policy status	SPDX ID	License Name	Obligations	Action
Always allowed	0BSD	BSD Zero Clause License	0	Details
No reviewed yet	AAL	Attribution Assurance License	Unknown	Details
No reviewed yet	Abstyles	Abstyles License	Unknown	Details
Never allowed	Adobe-2006	Adobe Systems Incorporated Source Code License Agreement	2	Details
No reviewed yet	Adobe-Glyph	Adobe Glyph List License	Unknown	Details
Always allowed	ADSL	Amazon Digital Services License	0	Details
Allowed depending on context	AFL-1.1	Academic Free License v1.1	6	Details
Always allowed	AFL-1.2	Academic Free License v1.2	7	Details
Always allowed	AFL-2.0	Academic Free License v2.0	7	Details
Always allowed	AFL-2.1	Academic Free License v2.1	7	Details

We define a core set of generic obligations

- It is sometimes more efficient to follow the same processes than to stick to the bare minimum of required obligations (e.g. add the licence in the documentation, even if it’s a BSD0)
- These generic obligations can be gathered in a “core set”
- This allows to see only actions that would need specific attention

What we do (ﾉ)



We ingest SBOMs

- Currently we support
  - a specific ORT format (EvaluatedModel), which is very thorough and includes the notion scopes and sub-projects
  - SPDX (partially tested)
- We plan to support Cyclone DX soon
- During ingestion it is possible to specify the type of technical relation between the 3rd party component and your own code base

We validate SBOMS

- In 5 steps:
  1) Presence of valid SPDX licence expression
  2) All licences have been reviewed by the legal department
  3) “AND”s are actual “AND”s and not “OR”s
  4) Choices (e.g. “MIT OR GPL-2.0-only”) have been decided
  5) Licences are compliant to the organisation’s policy
- We handle generalisation of decisions

We handle exploitation choices

- For each scope/subproject, you can indicate the type of exploitation that will be made of it
- The exploitation can be also set on a per component usage inside a release

We calculate resulting obligations

- Combining the information attached to the BOM and the qualification of the licences, we calculate the obligations that have to be followed for the release of your product to be compliant.

List of generic obligations to follow

In core	Generic name	Lead	Metacategory	Passivity
In core	License and copyright notices in documentation	DevQA	Mentions	Active
In core	No use of names for endorsement	Communication	Communication	Passive
In core	Preserve IP mentions in Source code	DevQA	Mentions	Active
In core	Patent Peace	Legal	IPManagement	Passive
Not in core	Weak Copyleft	Legal	IPManagement	Active
Not in core	Providing CSC to end user	DevQA	ProvidingSourceCode	Active
In core	Respect trademarks	Communication	Communication	Passive
Not in core	Indemnification of contributors	Legal	LicenseAgreement	Active
Not in core	License Agreement must exclude other contributors for additionnal terms	Legal	LicenseAgreement	Active

We keep track of the usages of the components

- As every validated BOM is stored in a DB, it’s easy to know the releases of a product containing a given version of a FOSS component
- Metadata for components are populated from scans (for ORT imports)

3rd Party Components

Top 10 Most used components

Name	Number of usages	Description
cache	4	PHP Doctrine Cache library is a popular c...
event-dispatcher	4	Standard interfaces for event handling.
python-dateutil	3	Extensions to the standard Python datetim...
idna	3	Internationalized Domain Names in Applica...
xmlschema	3	An XML Schema validator and decoder
defusedxml	3	XML bomb protection for Python stdlib mod...
zipp	3	Backport of pathlib-compatible object wra...
urllib3	3	HTTP library with thread-safe connection ...
six	3	Python 2 and 3 compatibility utilities
cffi	3	Foreign Function Interface for Python cal...

All components

Search | Search | page 1 of 12 | next | last

Name	Versions	Description

event-dispatcher

Description

Standard interfaces for event handling.

Versions

event-dispatcher : 1.0.0

The license expression is : MIT .

This component is used in the following releases of your products:

- Produit de test : 1.0.1
- Produit de test : 1.0.1

event-dispatcher : v6.0.3

Edit this version here :

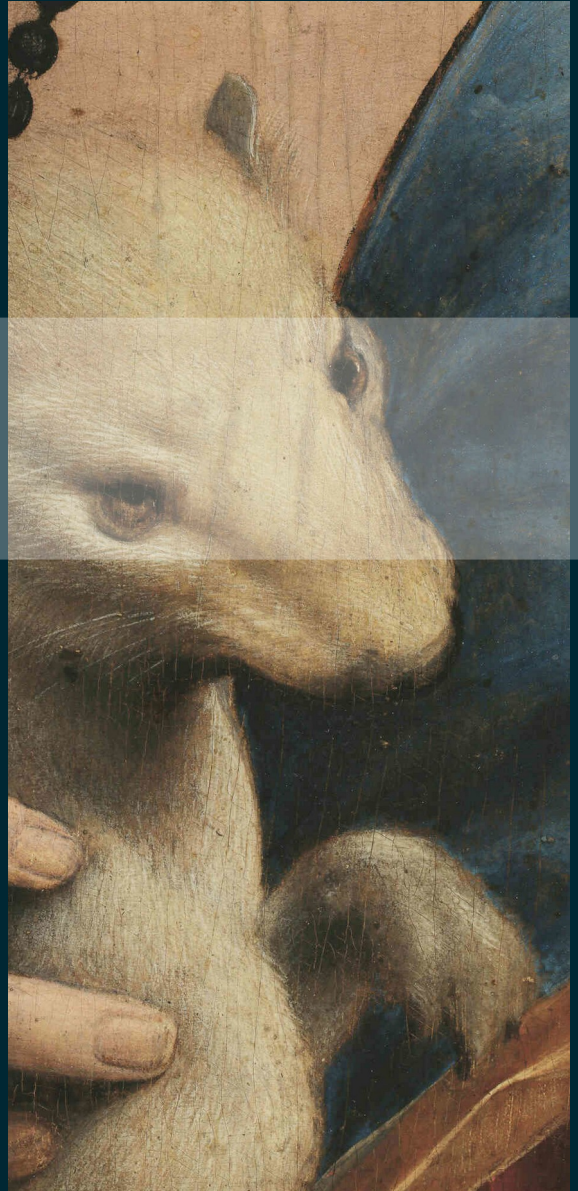
The license expression is :

MIT .

This component is used in the following releases of your products:

- Produit de test : 1.0.1
- Produit de test : 1.0.1

How we do it



We have a REST API and a Web UI

- The web UI is convenient for one-off operations, like 3rd party audits
- REST API is key for most case, where integration in the CI is mandatory

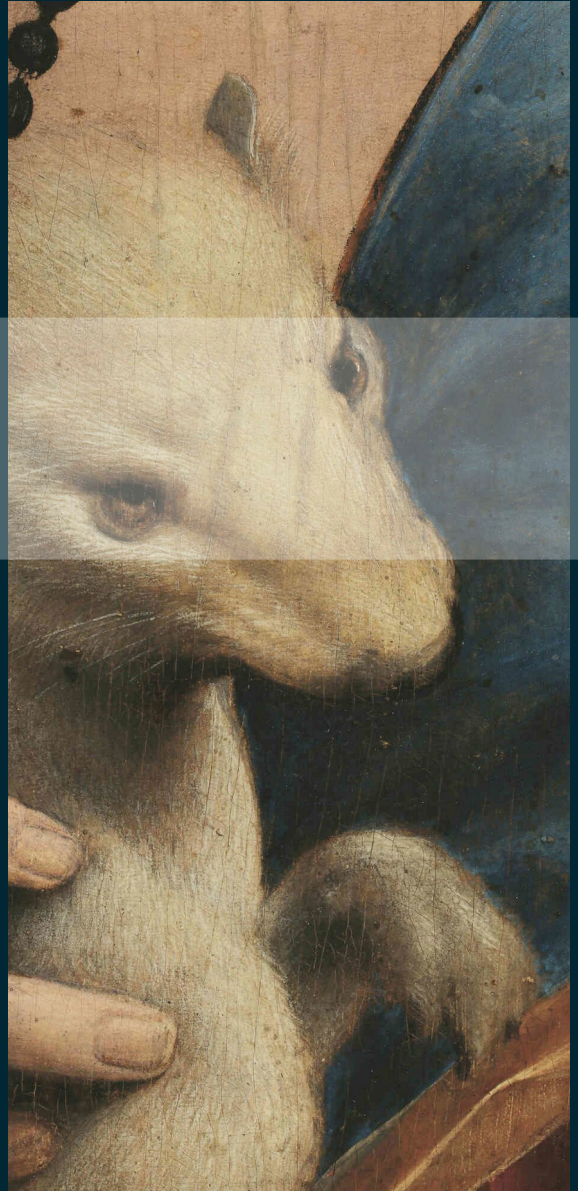
The stack we use

- Python / Django / Django REST / Bulma CSS
- No JS framework at the moment
- Currently database agnostic, but PostgreSQL preferred
- Deployment through Docker with Caddy / Gunicorn

We work in the open

- The code is available under the AGPL-3.0-only license at:
  - https://gitlab.com/hermine-project/hermine
- The documentation is available under the CC-BY-4.0 license at:
  - https://docs.hermine-foss.org/

Future



Next steps

- Publication of a V1 of the code by the end of the year
- Stabilize the data model for licences / obligations and select a license
- Publication of V1 of the dataset