On our way to (scholarly ?) edition

Implementing vectorization in EiDA : how and why ?

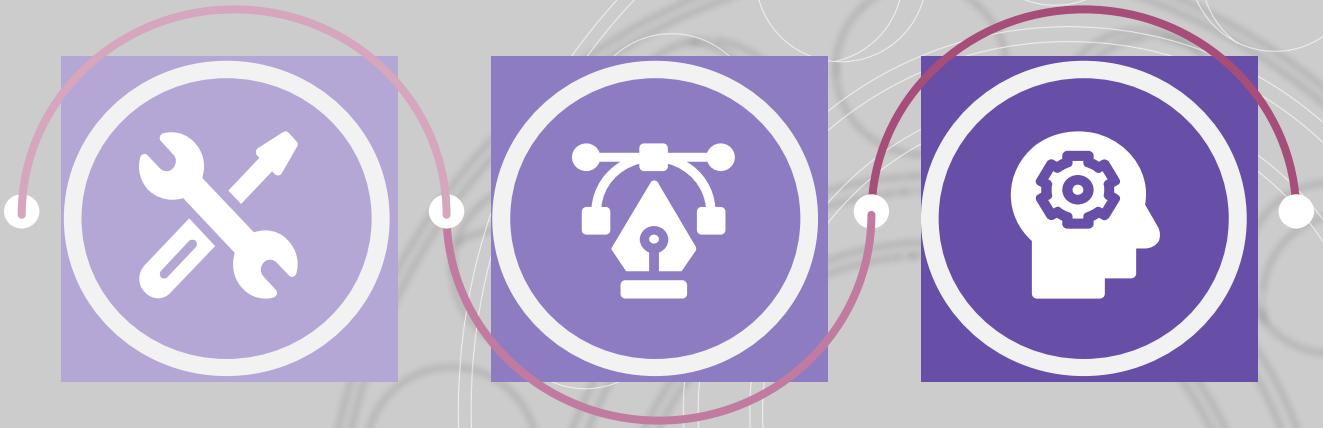
July 23th, 2024

01

The vectorization task in EiDA : how does it work ?

03

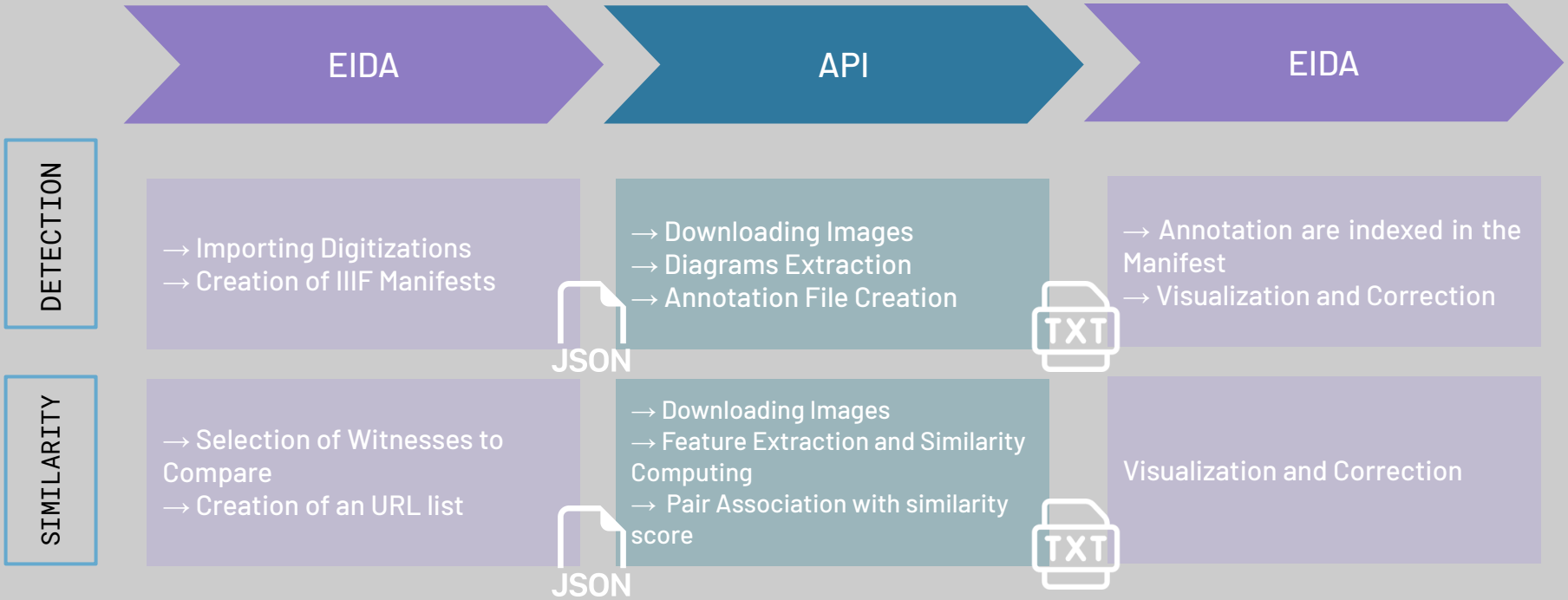
The base for scholarly edition of diagrams ?

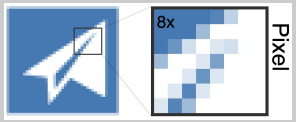


02

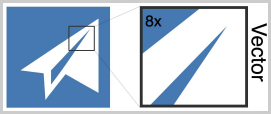
Formats and layers

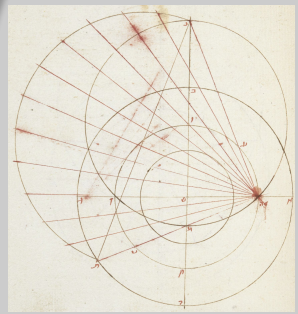
Just for context: what are the tools already implemented in EiDA ?

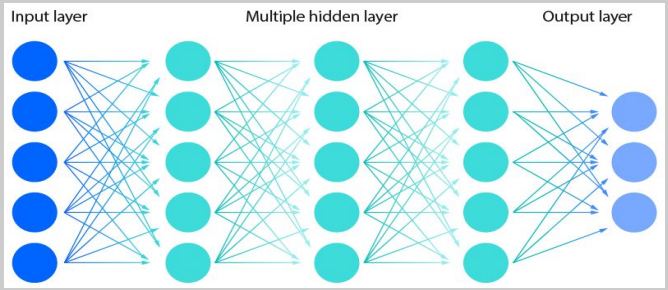


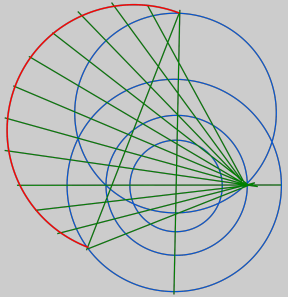


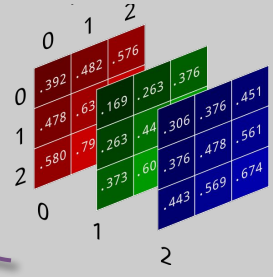
Task
Vectorization











Using deep neural network to output geometric shapes or *primitives*

p_{center1}, r_1
p_{center2}, r_2
p_{center3}, r_3
p_{center4}, r_4
$p_{\text{start1}}, p_{\text{end1}}$
$p_{\text{start2}}, p_{\text{end2}}$
.
.
$p_{\text{start14}}, p_{\text{end14}}$
$p_{\text{start1}}, p_{\text{mid1}}, p_{\text{end1}}$

Extraire

wit5_man20_0139_161,1511,896,831.npz

Emplacement : /

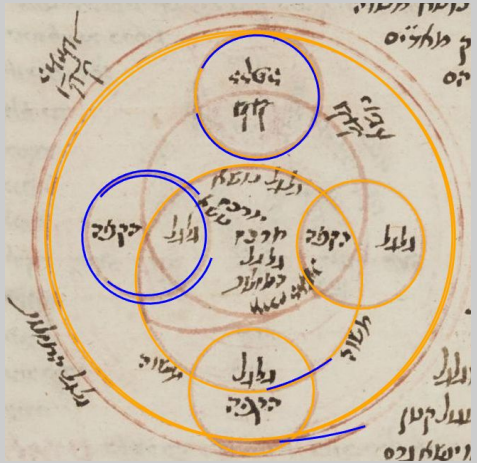
Nom	Taille	Type	Modifié
arcs.npy	176 octets	inconnu	01 janvier 1980, 00:00
arc_scores.npy	132 octets	inconnu	01 janvier 1980, 00:00
circles.npy	200 octets	inconnu	01 janvier 1980, 00:00
circle_scores.npy	140 octets	inconnu	01 janvier 1980, 00:00
lines.npy	256 octets	inconnu	01 janvier 1980, 00:00
line_scores.npy	144 octets	inconnu	01 janvier 1980, 00:00

```python
from numpy import load

data = load('wit5_man20_0139_161,1511,896,831.npz')
lst = data.files
for item in lst:
    print(item)
    print(data[item])
```

```
lines
[[134.81066895 457.17818451 779.38134766 248.8579483 ]
 [450.41806698 701.88641357 445.5779047   29.55957031]
 [ 82.28451538 235.88547516 751.17868042 460.67421722]
 [846.97953796 687.73287964 839.02302551  46.39743042]]
line_scores
[0.9833361  0.97366863 0.91139853 0.6446364 ]
circles
[[444.50769043 358.36816406 325.14129639]
 [122.31943512 357.18722534  99.33002472]
 [771.32922363 357.23934937 100.2789917 ]]
circle_scores
[0.9783766 0.955871  0.8885103]
arcs
[[720.67673302 494.20816517 745.99196815 474.5509901  740.33850098
  491.46079016]]
arc_scores
[0.31675324]
```

```
<?xml version='1.0' encoding='us-ascii'?>
<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" baseProfile="tiny" height="645" version="1.2" width="661"
xmlns:inkscape="http://www.inkscape.org/namespaces/inkscape" xmlns:sodipodi="http://sodipodi.sourceforge.net/DTD/sodipodi-0.dtd" inkscape:version="1.3
(0e150ed, 2023-07-21)">
  <defs />
  <image height="645" width="661" x="0" xlink:href="ms156_0161_706,383,661,645.jpg" y="0" />
  <circle cx="345.28564" cy="386.82678" fill="none" r="159.26169" stroke="orange" stroke-width="3" />
  <circle cx="502.60535" cy="339.225" fill="none" r="89.67699" stroke="orange" stroke-width="3" />
  <circle cx="357.63202" cy="129.9657" fill="none" r="87.40253" stroke="orange" stroke-width="3" />
  <circle cx="349.0131" cy="545.8338" fill="none" r="87.56171" stroke="orange" stroke-width="3" />
  <circle cx="350.35507" cy="328.02127" fill="none" r="284.86337" stroke="orange" stroke-width="3" />
  <circle cx="345.2689" cy="326.3235" fill="none" r="286.5298" stroke="orange" stroke-width="3" />
  <path d="M 388.352112,615.951721 A 516.823365,516.823365 0.000000 0,0 509.149384,594.206848" fill="none" stroke="blue" stroke-width="3"
sodipodi:type="arc" sodipodi:arc-type="arc" sodipodi:cx="357.8361831194576" sodipodi:cy="100.03005185391658" sodipodi:rx="516.8233649999999"
sodipodi:ry="516.8233649999999" sodipodi:start="1.273668118159911" sodipodi:end="1.511716783813232" />
  <path d="M 272.026550,155.109222 A 87.436038,87.436038 0.000000 1,0 279.412476,91.624451" fill="none" stroke="blue" stroke-width="3" sodipodi:type="arc"
sodipodi:arc-type="arc" sodipodi:cx="356.56123950106826" sodipodi:cy="132.7721001599907" sodipodi:rx="87.436038" sodipodi:ry="87.436038"
sodipodi:start="3.6315665495298615" sodipodi:end="2.883260775972637" />
  <path d="M 136.343063,272.347351 A 84.086329,84.086329 0.000000 1,1 146.534378,390.720276" fill="none" stroke="blue" stroke-width="3" sodipodi:type="arc"
sodipodi:arc-type="arc" sodipodi:cx="200.72995024243303" sodipodi:cy="326.42913606601326" sodipodi:rx="84.08632899999999" sodipodi:ry="84.08632899999999"
sodipodi:start="3.840223336568032" sodipodi:end="2.2711955657458702" />
  <path d="M 275.689087,268.669250 A 93.344812,93.344812 0.000000 1,0 292.839874,357.897491" fill="none" stroke="blue" stroke-width="3" sodipodi:type="arc"
sodipodi:arc-type="arc" sodipodi:cx="204.18709960784327" sodipodi:cy="328.6752474193181" sodipodi:rx="93.34481200000002" sodipodi:ry="93.34481200000002"
sodipodi:start="0.3184100906483458" sodipodi:end="5.584982158241335" />
  <path d="M 370.369690,542.614502 A 265.937090,265.937090 0.000000 0,0 461.814789,499.437164" fill="none" stroke="blue" stroke-width="3"
sodipodi:type="arc" sodipodi:arc-type="arc" sodipodi:cx="304.6175623541661" sodipodi:cy="284.93409229116304" sodipodi:rx="265.93709"
sodipodi:ry="265.93709" sodipodi:start="0.9383659635209984" sodipodi:end="1.3209583965330074" />
</svg>
```



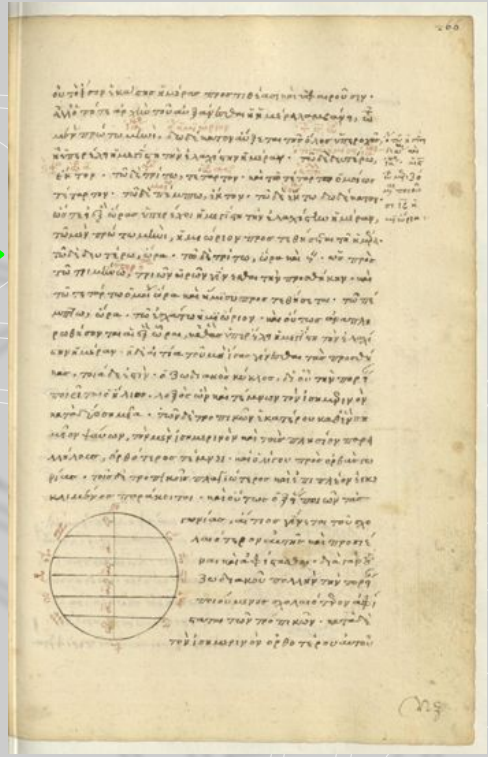
01

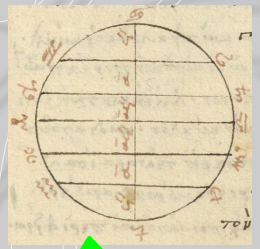
The vectorization task in EiDA : how does it work ?

A reminder about IIIF

Provides a standard framework for accessing images thanks to human hackable URLs

https://eida.obspm.fr/iiif/2/wit9_img9_0533.jpg/full/full/0/default.jpg





{scheme}://{server}{/prefix}/{identifier}/{region}/{size}/{rotation}/{quality}.{format}

https://eida.obspm.fr/iiif/2/wit9_img9_0533.jpg/80,1658,524,515/full/0/default.jpg

JSON to communicate with the API


```
{
  "doc_id":"wit14_img14_anno14",
  "model_name" : "0036",
  "images": {
    "wit14_img14_0029_255,363,341,614":
"https://eida.obspm.fr/iiif/2/wit14_img14_0029.jpg/255,363,341,614/full/0/default.jpg",
    "wit14_img14_0029_604,355,313,379":
"https://eida.obspm.fr/iiif/2/wit14_img14_0029.jpg/604,355,313,379/full/0/default.jpg",
    "wit14_img14_0030_250,393,540,587":
"https://eida.obspm.fr/iiif/2/wit14_img14_0030.jpg/250,393,540,587/full/0/default.jpg",
    "wit14_img14_0033_627,426,327,429":
"https://eida.obspm.fr/iiif/2/wit14_img14_0033.jpg/627,426,327,429/full/0/default.jpg",
    "wit14_img14_0033_231,393,382,427":
"https://eida.obspm.fr/iiif/2/wit14_img14_0033.jpg/231,393,382,427/full/0/default.jpg",
    "wit14_img14_0035_299,352,332,350":
"https://eida.obspm.fr/iiif/2/wit14_img14_0035.jpg/299,352,332,350/full/0/default.jpg",
    …
  }
}
```

Storage of Digitizations

Syrte9 Server

Data Base

Images

django

Application EIDA

JSON : a list of URLs (IIIF Image API)

HTTP Requests

Dishas-ia GPU

ultralytics YOLOv5

SegSwap

HDV (vectorization)

API + Models

Visualization of Predictions

SVG Files

local machine

Images Download (deletion after two weeks)

+

Model Inference

Workflow

@dramatiq.task

RETRIEVE URLS CORRESPONDING TO PAGE REGIONS / DIAGRAMS EXTRACTED

SENDS JSON DATA TO THE API

CHECKS THE RECEIVED DATA AND PARSES IT

IMAGES ARE DOWNLOADED

MODEL INFERENCE

ZIP SVG FILES - SENT TO EIDA

ENDPOINT (RECEPTION OF THE ZIP FILE)

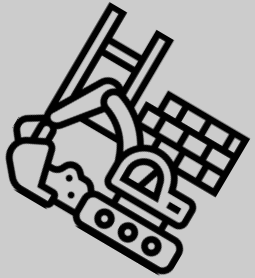
UNZIPS, OPENS, READS AND WRITES SVG FILES ON EIDA

VIZ

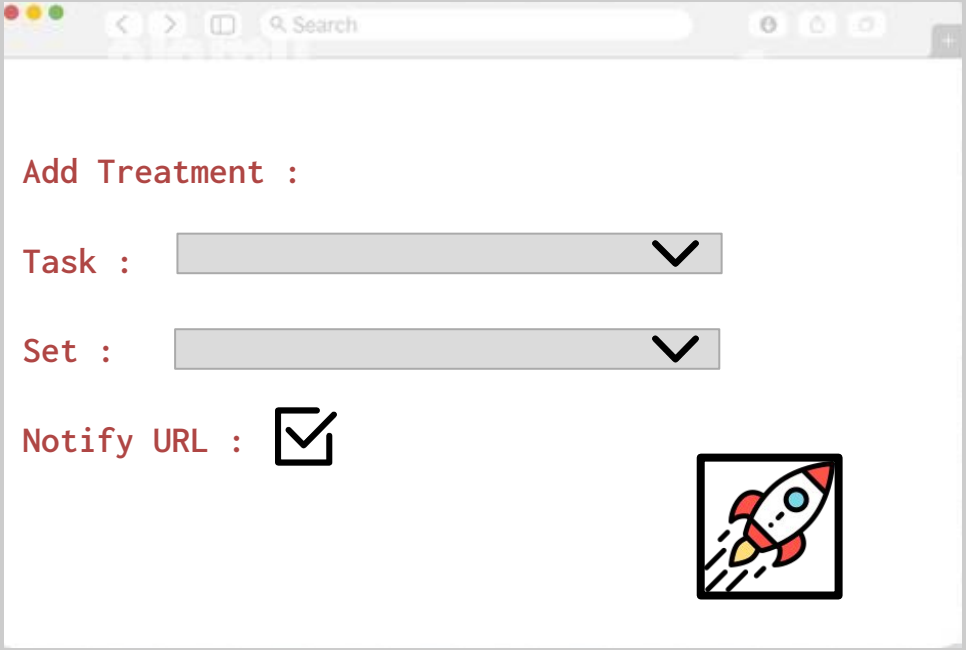
?

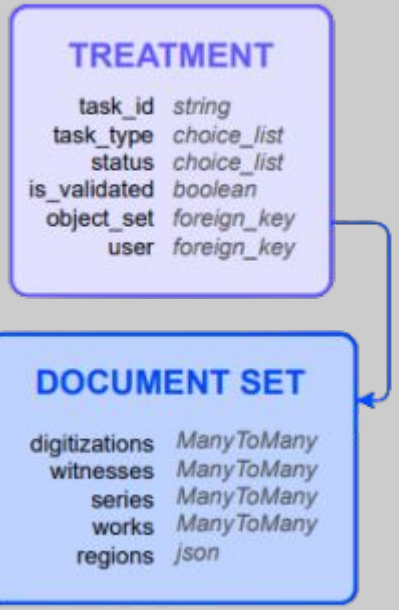
A Modular Platform

→ More flexible
→ User can custom
→ Facilitate management and maintenance

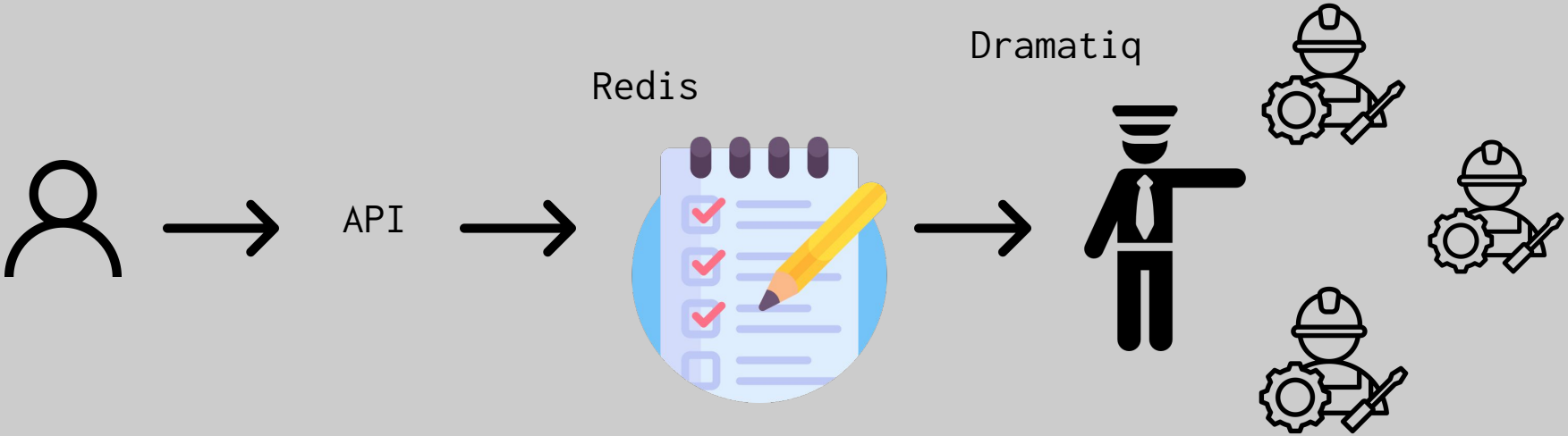


A Modular Platform

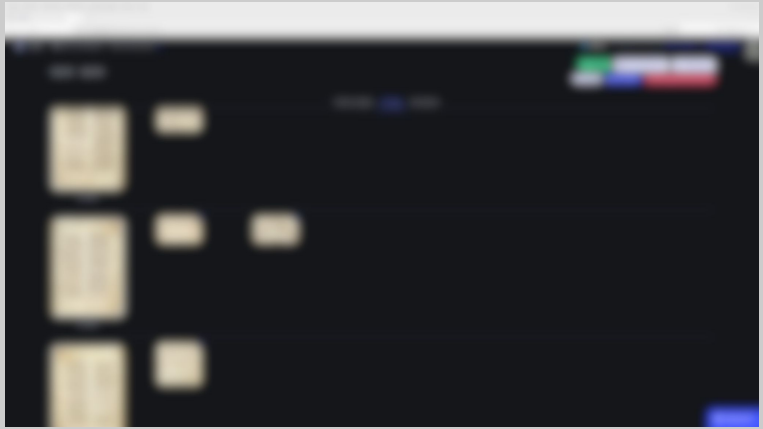


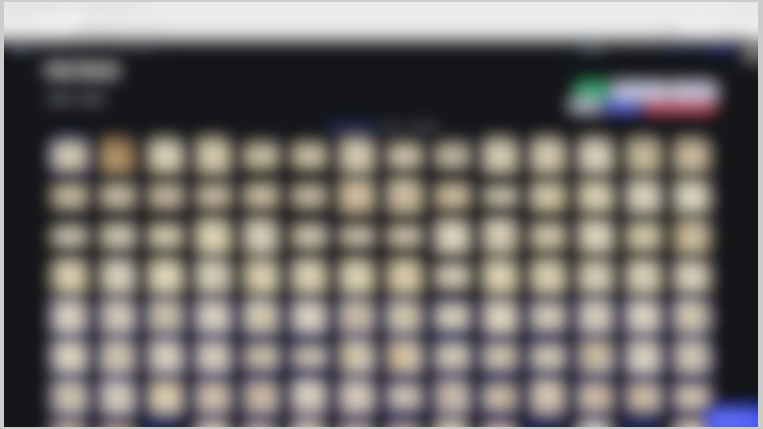


Task Management in the API



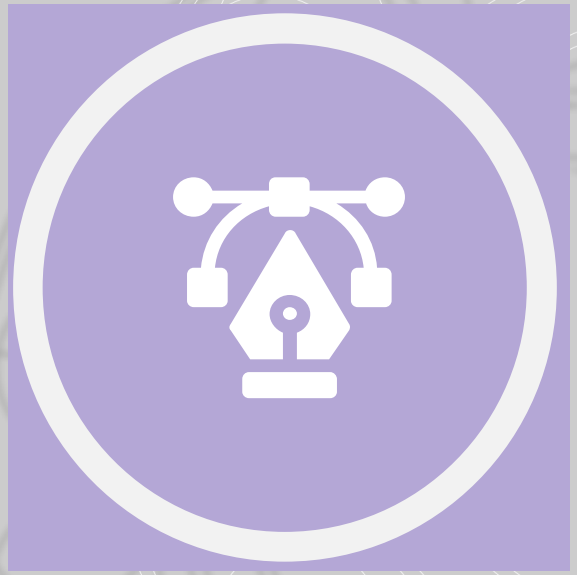
Visualize the Results





02

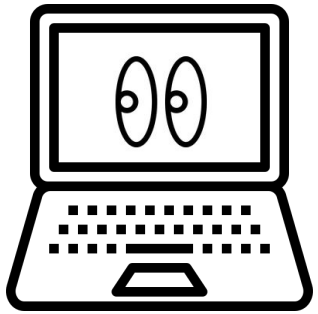
Layers and formats



Why SVGs ?

Embed metadata

```
<?xml version='1.0' encoding='us-ascii'?>
<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" baseProfile="tiny" height="1376" version="1.2" width="1368"
xmlns:inkscape="http://www.inkscape.org/namespaces/inkscape" xmlns:sodipodi="http://sodipodi.sourceforge.net/DTD/sodipodi-0.dtd" inkscape:version="1.3 (0e150ed, 2023-07-21)">
  <defs />
  <image height="1376" width="1368" x="0" xlink:href="ms155_0056_178,878,1368,1376.jpg" y="0" />
  <circle cx="679.79" cy="696.2864" fill="none" r="627.1144" stroke="orange" stroke-width="3" />
  <circle cx="688.0594" cy="503.96643" fill="none" r="434.87476" stroke="orange" stroke-width="3" />
  <circle cx="680.7131" cy="687.0875" fill="none" r="277.15222" stroke="orange" stroke-width="3" />
  <circle cx="671.986" cy="511.75793" fill="none" r="445.23505" stroke="orange" stroke-width="3" />
  <circle cx="682.8999" cy="688.596" fill="none" r="421.30588" stroke="orange" stroke-width="3" />
  <circle cx="689.08167" cy="506.3659" fill="none" r="421.4337" stroke="orange" stroke-width="3" />
  <path d="M 256.28815 688.8925 L 1123.9667 490.76904" fill="none" stroke="green" stroke-width="3" />
  <path d="M 670.82275 54.234756 L 690.5636 1321.8917" fill="none" stroke="green" stroke-width="3" />
  <path d="M 255.44974 688.98755 L 1093.156 594.4521" fill="none" stroke="green" stroke-width="3" />
  <path d="M 631.76855 666.5324 L 1122.2913 432.85513" fill="none" stroke="green" stroke-width="3" />
  <path d="M 653.0232 620.9497 L 1001.2896 194.13893" fill="none" stroke="green" stroke-width="3" />
  <path d="M 498.3139 992.54193 L 724.8771 572.5667" fill="none" stroke="green" stroke-width="3" />
  <path d="M 311.59433 908.2044 L 579.469 687.52496" fill="none" stroke="green" stroke-width="3" />
  <path d="M 532.077698,881.705505 A 385.647756,385.647756 0.000000 1,1 978.377686,780.289124" fill="none" stroke="blue" stroke-width="3" sodipodi:type="arc" sodipodi:arc-type="arc"
sodipodi:cx="686.4433171904819" sodipodi:cy="528.300005156301" sodipodi:rx="385.6477560000001" sodipodi:ry="385.6477560000001" sodipodi:start="1.9826145540847406"
sodipodi:end="0.7120906668284839" />
</svg>
```



Computable geometrical content

Why SVGs ?

shape

coordinates

```
<path d="M 93.79881286621094 108.4345703125 L 480.60060119628906 448.5714111328125" fill="none"
stroke="green" stroke-width="3" />
```

color of the stroke (now depending on the shape predicted and not on the actual color on the manuscript)

thickness of the stroke (now by default)

the shape is filled or not

```
<path d="M 274.125072,252.560776 A 261.320484,261.320484 0.000000 0,1 363.440845,473.154739" fill="none"
stroke="firebrick" stroke-width="3" sodipodi:type="arc" sodipodi:arc-type="arc"
sodipodi:cx="103.13299192417642" sodipodi:cy="450.1717673677308" sodipodi:rx="261.32048399999996"
sodipodi:ry="261.32048399999996" sodipodi:start="5.425696637289944" sodipodi:end="0.0880631480620444" />
```

Why SVGs ?

Computable

- ➔ The machine understands what geometric content is drawn.
- ➔ Allows to analyse and manipulate the geometric content of the diagram.

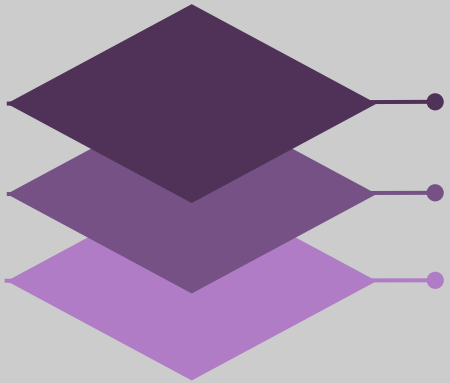
Similarity Search

- ➔ Perform Similarity Search among a large database of diagrams.
- ➔ Allows to find similar diagrams even if the actual image doesn't look the same.
- ➔ Allows to search through a batch of Diagrams those that have the same set of geometric shape, or retrieve one specific pattern.

Edition

- ➔ Possible to correct the prediction
- ➔ A base for different kinds of editions

One Diagram



JPG

SVG

XML

?

?

?

Edge Detection

Similarity (cosine + SegSwap)

Classification

#color

#graduations

geometric shape + color + thickness

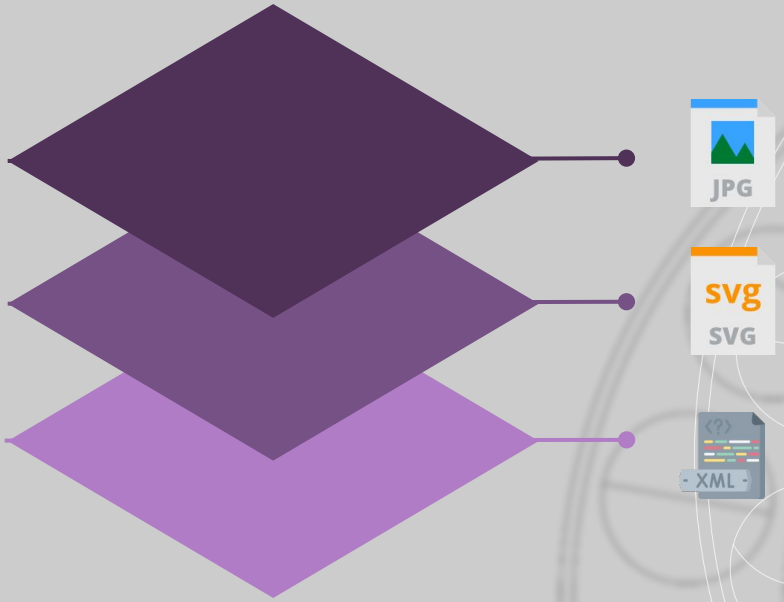
Sim . based on geometrical content

Edition

labels

coordinates + transcription ? (ALTO)

One Diagram



Requirements

- Make it coexist (data model);
- Ability to navigate through the thickness of a single image, allowing for toggling and alignment of different layers.
- Implement advanced search functions based on metadata derived from the manuscript witnesses or from classifications/tagging at the diagram level, or from mining in computable formats.
- ***Create different types of edition of a diagram***

03

A base for a scholarly edition ?

A Tool and a Standardized Method

GOALS

➔ **Create an Editing Tool:**
- Extensive SVG editing platform to manipulate SVGs (model prediction correction + manual tracing).
- Different types of edition.

➔ **Develop a Comprehensive Method:**
- Ensure epistemological and historiographic accuracy.
- Draw on interdisciplinary collaboration (history, semiotics, media studies, digital humanities).
- Use automatic vectorization and SVG format to embed attributes documenting the editing process.
- Adapt to diverse perspectives, including philological and variationist points of view.

Three Types of Edition

	DIPLOMATIC	CRITICAL, CORRECTED
Which object do we modify	Edge Detection (JPEG) OR SVG + Labels	New SVG object !
It requires ...	➔ JPEG image (eventually with external modifications) OR correct on the SVG file the defects of the automatic transcription. ➔ Eventually embed a confidence level + ?; ➔ Decide on a state of the image that is no longer just a correction but a diplomatic edition.	➔ A critical apparatus (X witnesses of the same diagram) ➔ Draw a new version of this diagram ➔ Select the pattern / significant element you want to highlight and perform automatic collation on the critical apparatus + validate manually. ➔ Visualize the sources linked to each significant element; ➔ Interact with it: switch between representations of the sources, place it in transparency etc. ➔ If corrected edition add more metadata linking the textual proposition?

Scholarly Edition

New SVG object !

→ A critical apparatus (X witnesses of the same diagram);

→ Draw your own version of this diagram;

→ Select the patterns / significant elements you want to highlight and perform automatic collation + validate manually;

→ Visualize the sources linked to each significant element ;

→ Interact with it: switch between representations of the sources, place it in transparency etc.

Show invariants instead of variants:

- ➔ Consider and visualize all witnesses, showing sources for each editing decision.
- ➔ Critical edition as one type of scholarly edition.

Interactive:

- ➔ An edition to visit a corpus of images.
- ➔ Navigate through different representations of the diagrams (sandbox ?).

A Method:

- ➔ To document visually a scientific hypothesis.
- ➔ Thanks to computable formats.

Question: Is it even possible to print it? What are the modalities of **sharing**?

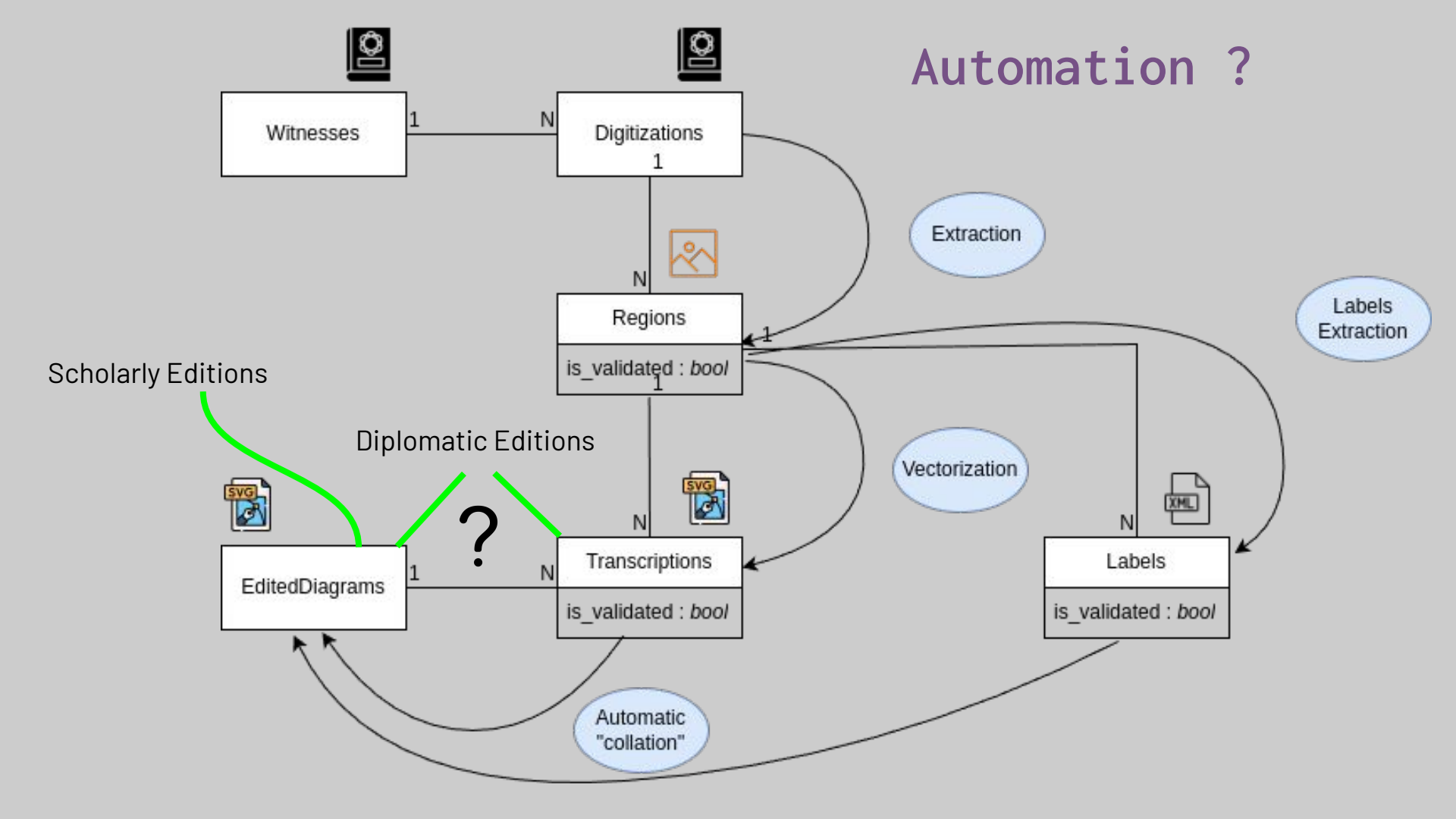
Three blocks

DRAWING

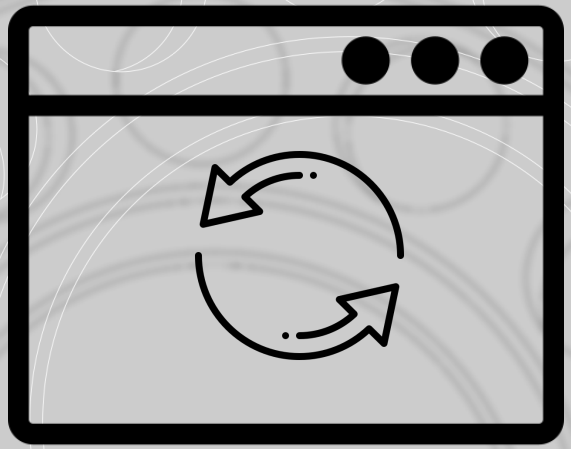
COLLATION

VISUALIZATION

Automation ?


Witnesses
1
N
Digitizations
1
N
Regions
is_validated : bool
1
1
Extraction
Labels Extraction
Scholarly Editions
Diplomatic Editions
?
SVG
SVG
XML
N
N
EditedDiagrams
1
N
Transcriptions
is_validated : bool
Vectorization
Labels
is_validated : bool
Automatic "collation"

Backend vs. Frontend



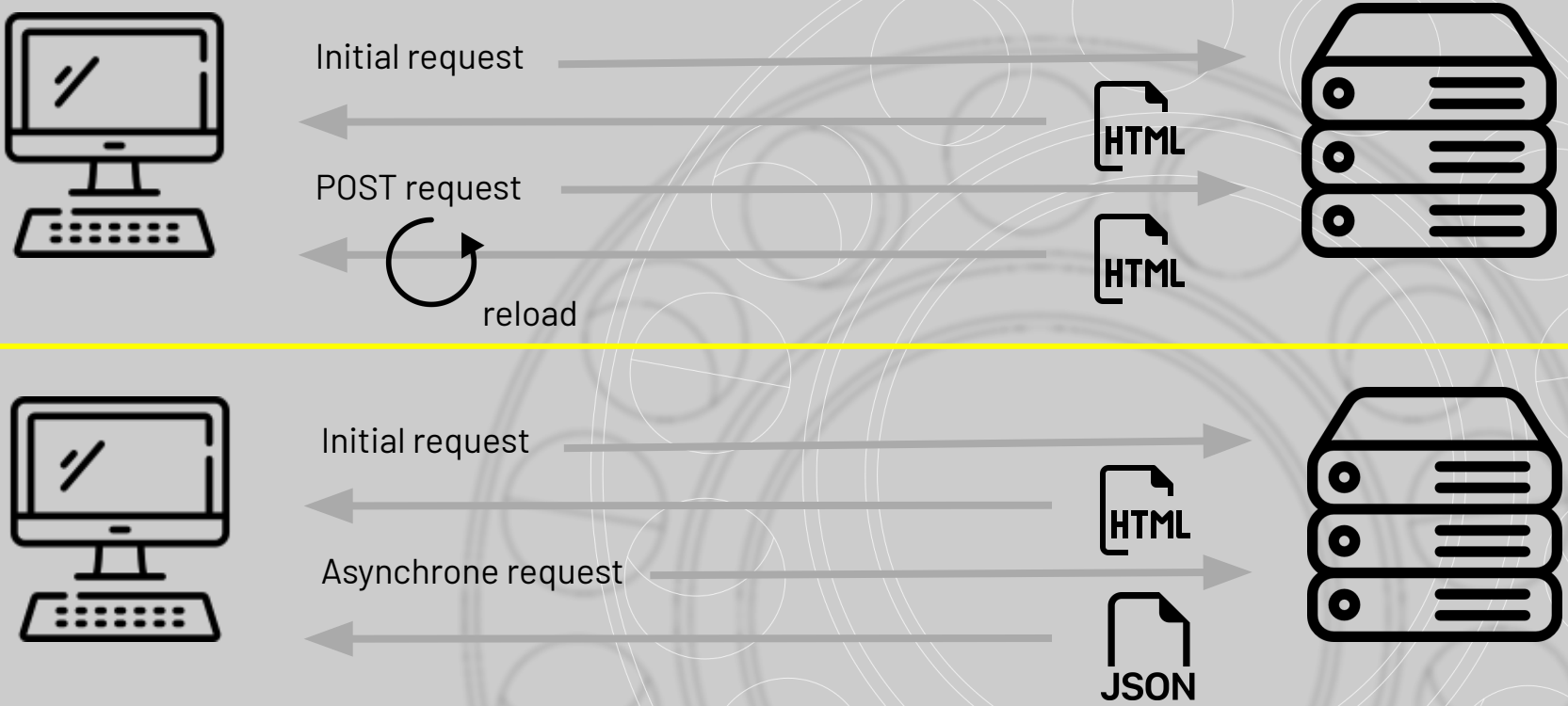
Backend vs. Frontend

→ New Object in the DataBase

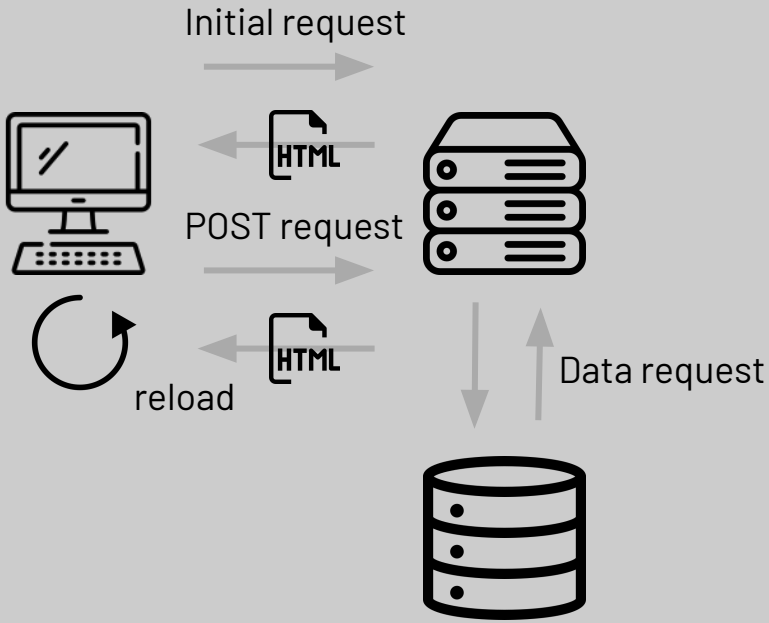
→ Server Communication

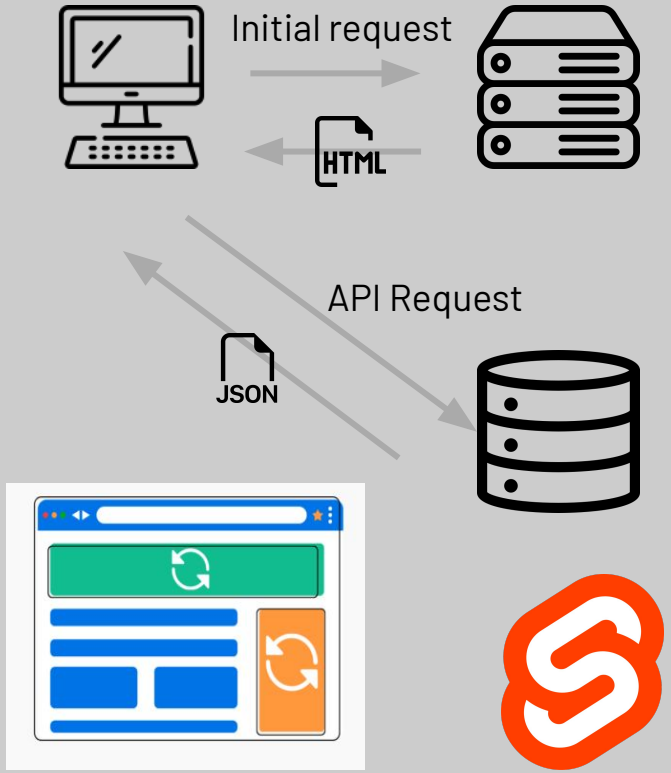
→ Concurrency ?

Communication with the server: MPA vs. SPA



Communication with the server: MPA vs. SPA





Data Management : many questions

New Object in the Database :

It requires modeling the diagram as an ordered hierarchy of units with attributes.

EDITED_DIAGRAM

ID	*int*
author	*string*
type	*foreign_key*
astronomical_objects	*ManyToMany*
is_complete	*boolean*
date	*int*
sources	*json*
lines	*json*
circles	*json*
arcs	*json*
groups	*json*

➔ Define the level of granularity on which we want to perform the automatic collation?

→ a layer of content (the whole pattern / all the labels)?

→ one geometrical or textual component?

→ which attributes become filters?

➔ For a corrected edition: would you need to embed more metadata concerning the proposition it relies on? A link to the textual content?

➔ In general what kind of metadata/classification/manual tagging do you need to add to your diagrams to => become new filters for the similarity search?