europeana sounds

Project Acronym:	**Europeana Sounds**
Grant Agreement no:	**620591**
Project Title:	**Europeana Sounds**

D2.10 Development of the Crowdsourcing Infrastructure

Revision: Final

Date: 31/12/2015

Authors: Remy Gardien, Vassilis Tzouvaras, Maarten Brinkerink, Hugo Manguinhas, Sergiu Gordea, Alessio Piccioli, Lise Schauer, Antoine Isaac, Breandán Knowlton

Abstract: This document reports on the status of the development of the crowdsourcing infrastructure for Europeana Sounds. All technical partners present an account of their development progress, provide details on the implementation and share a forecast of the development and steps ahead to work towards a final infrastructure as specified earlier in Work Package 2.

Dissemination level	
Public	X
Confidential, only for the members of the Consortium and Commission Services	

Coordinated by the British Library, the Europeana Sounds project is co-funded by the European Union, through the ICT Policy Support Programme as part of the Competitiveness and Innovation Framework Programme (CIP) http://ec.europa.eu/information_society/activities/ict_psp/

Revision history

Version	Status	Name, organisation	Date	Changes
0.1	ToC	Remy Gardien, Hugo Manguinhas, EF; Sergiu Gordea, AIT; Alessio Piccioli, NET7; Vassilis Tzouvaras, NTUA; Lise Schauer, HP; Maarten Brinkerink, NISV	07/12/2015	First table of content, based on existing documentation created during the development process
0.5	1st draft	Remy Gardien, Hugo Manguinhas, Antoine Isaac, EF; Sergiu Gordea, AIT; Alessio Piccioli NET7; Vassilis Tzouvaras, NTUA; Lise Schauer, Breandán Knowlton, HP; Maarten Brinkerink, NISV	14/12/2015	First draft, extending the 0.1 version of the document to a more coherent and detailed report
0.8	2nd draft	Remy Gardien, Hugo Manguinhas, Antoine Isaac, EF; Sergiu Gordea, AIT; Alessio Piccioli, NET7; Vassilis Tzouvaras, NTUA; Maarten Brinkerink, NISV	18/12/2015	Near-final version, resulting from several internal review rounds and copy editing
0.9	Final draft	Remy Gardien, EF; Vassilis Tzouvaras, NTUA; Maarten Brinkerink, NISV	28/12/2015	Final version, taking into account feedback from external reviewers
1.0	Final	Richard Ranft, BL	30/12/2015	Layout, minor text changes

Review and approval

Action	Name, organisation	Date
Reviewed by	Yann Le Franc, EUDAT.eu Mia Ridge, The British Library	22/12/2015 24/12/2015
Approved by	Coordinator and PMB	30/12/2015

Distribution

No.	Date	Comment	Partner / WP
1	30/12/2015	Submitted to the European Commission	BL/WP7
2	30/12/2015	Posted on Europeana Pro website	BL/WP7
3	30/12/2015	Distributed to project consortium	BL/WP7

Application area

This document is a formal output for the European Commission, applicable to all members of the Europeana Sounds project and beneficiaries. This document reflects only the author’s views and the European Union is not liable for any use that might be made of information contained therein.

Statement of originality

This document contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Project summary

Europeana Sounds is Europeana's 'missing' fifth domain aggregator, joining APEX (Archives), EUscreen (television), the Europeana film Gateway (film) and TEL (libraries). It will increase the opportunities for access to and creative re-use of Europeana's audio and audio-related content and will build a sustainable best practice network of stakeholders in the content value chain to aggregate, enrich and share a critical mass of audio that meets the needs of public audiences, the creative industries (notably publishers) and researchers. The consortium of 24 partners will:

- Double the number of audio items accessible through Europeana to over 1 million and improve geographical and thematic coverage by aggregating items with widespread popular appeal such as contemporary and classical music, traditional and folk music, the natural world, oral memory and languages and dialects.
- Add meaningful contextual knowledge and medium-specific metadata to 2 million items in Europeana's audio and audio-related collections, developing techniques for cross-media and cross-collection linking.
- Develop and validate audience specific sound channels and a distributed crowd-sourcing infrastructure for end-users that will improve Europeana's search facility, navigation and user experience. These can then be used for other communities and other media.
- Engage music publishers and rights holders in efforts to make more material accessible online through Europeana by resolving domain constraints and lack of access to commercially unviable (i.e. out-of-commerce) content.

These outcomes will be achieved through a network of leading sound archives working with specialists in audiovisual technology, rights issues, and software development. The network will expand to include other data-providers and mainstream distribution platforms (Historypin, Spotify, SoundCloud) to ensure the widest possible availability of their content.

For more information, visit http://pro.europeana.eu/web/europeana-sounds and http://www.europeanasounds.eu

Copyright notice

Copyright © Members of the Europeana Sounds Consortium, 2014-2017. This work is licensed under the Creative Commons CC-BY License: https://creativecommons.org/licenses/by/4.0/.

Contents

Executive summary: D2.10 Development of the Crowdsourcing Infrastructure

This document depicts the current status and an account of the development of the crowdsourcing infrastructure in Europeana Sounds. The infrastructure as it is implemented is based on the specifications as defined in WP2.

The general architecture of the infrastructure as a whole has been defined and is currently being implemented and further iterated on by the technical partners involved in this Work Package. The Annotations API (as developed by AIT), an automated programming interface developed on top of Europeana's API for developers, has the basis of its functionality implemented and will be released as an alpha version early in 2016 to start prototyping with. Developers and applications such as Europeana Collections will then be able to send in user annotations to Europeana via the API.

Historypin/Shift is working on a service to help traditional musicians identify and learn new tunes through listening to archival recordings. This service will be Tunepal, into which one can play a traditional Irish or Scottish tune and which will then surface archival recordings of that tune, pulled from the Europeana database. When a user selects a match between their performance or recording and the contributed archival objects, this connection is recorded in the form of an Open Annotation-compliant link, which can then be stored and later referenced.

NTUA progresses with developing the WITH platform, which is a culture-sharing crowdsourcing platform that allows Europeana Sounds partners, external Cultural institutions and third party developers to easily search for cultural resources with the aim to collect, enrich, share and co-create with other users. Everything users can do in WITH is saved as annotations, which could then later be synchronised to other applications such as the Annotations API.

NET7 is working on the Pundit widget, which is the main instrument of the widgets used in the context of Europeana Sounds. NET7 just released a new version of Pundit (Pundit 2) which has a more intuitive interface and is much more user-oriented. One of the next steps for the partners will be to integrate the Pundit annotations with the Annotations API.

1 Introduction

This document summarises the current status of the development of the crowdsourcing infrastructure in the Europeana Sounds project. The infrastructure as it is implemented is based on the specifications as defined in WP2. The infrastructure will be the basis for the implementation of crowdsourcing functionalities in various applications, such as Europeana Collections, in particularly the Music Channel. We report on work done so far, provide details on the technical implementation and an outline of the next steps for each of the individual components of the infrastructure. This report also evidences the reaching of project milestone MS29 *Crowdsourcing infrastructure*.[1]

Relevant documents that were completed before this deliverable and that provide meaningful context and reasoning on the implementation as described:

- D2.2 - Functional design of semantic enrichment [Ref 1]
- MS11 *Evaluation of first deployment of the crowdsourcing infrastructure* [Ref 2]

Europeana reports on the details of the core infrastructure, being the Europeana repository where all (crowdsourced) metadata lives and where in the end most of the crowdsourced metadata will reside.

AIT reports on the Annotations API, an automated programming interface build on top of the Europeana API which developers can use to annotate Europeana metadata. The section also covers some of the challenges and ongoing discussions within the implementation such as the topic of moderation and identification of annotations.

Historypin/Shift reports on the traditional music pilot, which focusses on the challenge of music identification. It describes the underlying user research, the proposed technical solution, and a timeline for the development and integration.

NTUA reports on the development of WITH, which is a culture-sharing crowdsourcing platform that allows Europeana Sounds partners, external Cultural institutions and third party developers to easily search for cultural resources with the aim to collect, enrich, share and co-create with other users.

And finally, NET7 reports on Pundit, a tool which Europeana Sounds data providers (or other developers) can integrate on their website to allow for annotations of metadata.

2 Overall system architecture

The basis of the Europeana Sounds crowdsourcing infrastructure has been implemented. All tools and components have first versions available and integrated. An updated system architecture can be seen below (Figure 1). This system architecture depicts the various tools and flows involved with supporting the various crowdsourcing scenarios. At the heart of the system architecture is the Annotations API, which implements a REST[2] (representational state transfer) interface for management and administration of annotations, a developer console used as documentation, and a testing and debugging environment (see also Section Annotations API). Applications such as the Europeana Collections [3],

[1] Note: This document reports on Project Deliverable D2.10. In the original Grant Agreement Annex 1 - Description of Work dated 17 October 2013, this deliverable was part of WP5 and numbered D5.4. In the new Grant Agreement Annex 1 - Description of Work dated 19 October 2015, the crowdsourcing infrastructure task is moved to task T2.5 in WP2, with its corresponding outputs, D2.10 and MS29.
[2] https://en.wikipedia.org/wiki/Representational_state_transfer
[3] http://www.europeana.eu

Pundit [4] and Tunepal [5] will connect with the Annotations API to allow their users to add annotations to Europeana objects. For applications unable to make a direct connection with the Annotations API, there is the alternative possibility of connecting to the so-called 'Round Tripping Daemon'[6] , which is able to harvest annotations from various sources, and import these asynchronously into Europeana, again utilising the Annotations API.

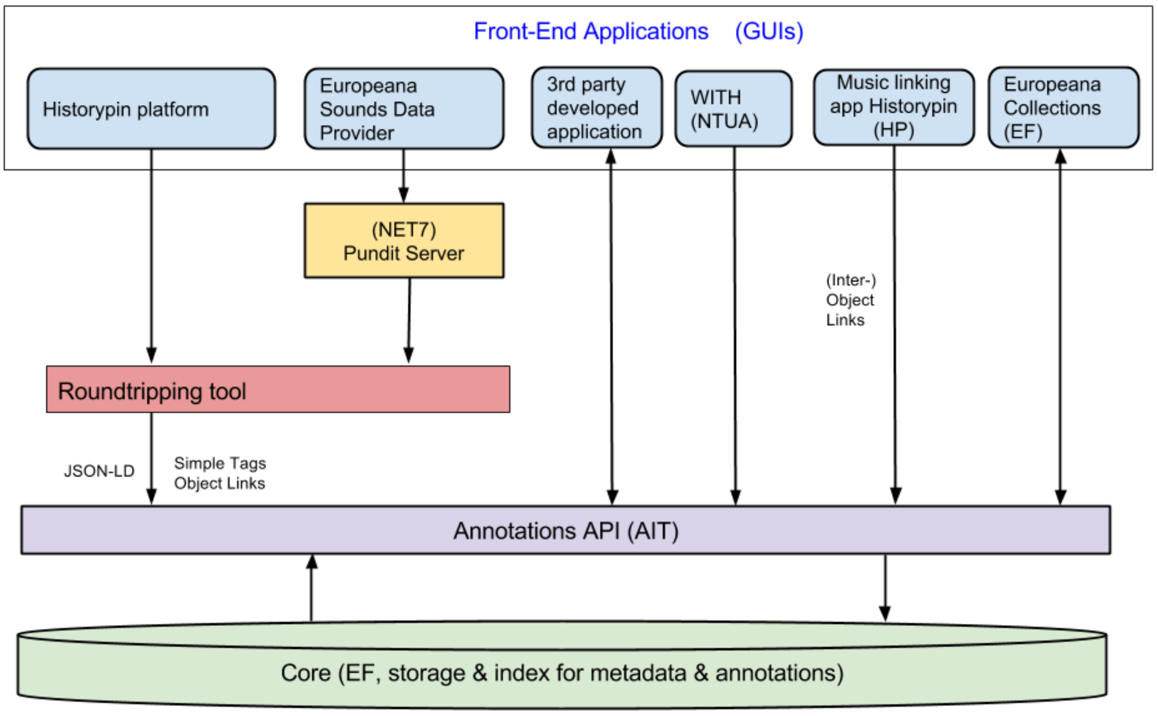

Front-End Applications (GUIs)
Historypin platform
Europeana Sounds Data Provider
3rd party developed application
WITH (NTUA)
Music linking app Historypin (HP)
Europeana Collections (EF)
(NET7) Pundit Server
(Inter-) Object Links
Roundtripping tool
JSON-LD
Simple Tags Object Links
Annotations API (AIT)
Core (EF, storage & index for metadata & annotations)

Figure 1: Updated system architecture Europeana Sounds

[4] http://thepund.it
[5] http://www.tunepal.org
[6] https://docs.google.com/document/d/1eP6CVFLtKqSCduhcuYnqPEKQR7_Za0cizNYE0SQ-6Og

2.1 Crowdsourcing infrastructure setup

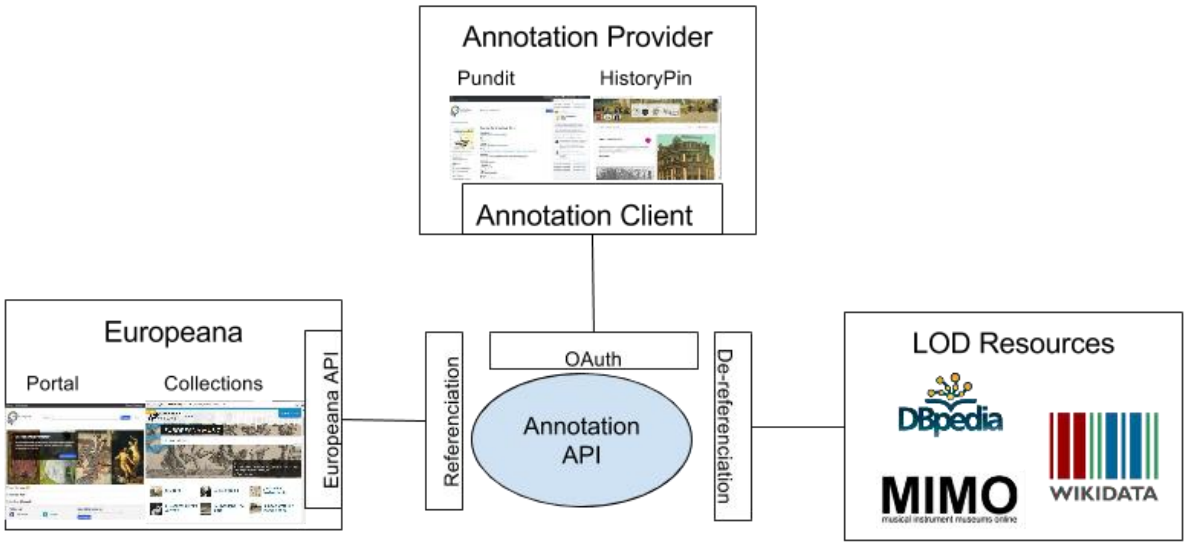


Figure 2: Conceptual overview of crowdsourcing infrastructure

Figure 2 above sketches the conceptual overview of the crowdsourcing infrastructure. This representation is correlated to the conceptual modelling of annotations, in which a user (i.e. creator) is providing additional information (i.e. body) to an existing internet resource (i.e. target). Following the same pattern, the users need to use a software application that collects their input (i.e. their annotation, or body) and transforms it into a standardized representation according to the Web Annotation Data Model (see Section 3.2.1. The data model). These software applications are referenced under the generic name of *Annotation Client*.

On behalf of end-users authenticated through the OAuth protocol,[7] clients submit requests to store annotations in the Core Infrastructure, represented by the *Annotation API*. The main goal of the Europeana Sounds Crowdsourcing Infrastructure is to provide an appropriate environment for supporting users to easily enrich Europeana items. Consequently, the target part of the annotations always references one or more objects available in Europeana. These objects are accessible to software applications through the Europeana API, and to the end users through Europeana Portal and Collections. Small pieces of information can be provided by users in textual form (i.e. in the case of tags or comments). However, semantic enrichments are often related to existing concepts that already have a rich description available on the web. *Linked Open Data Resources* are referenced through their HTTP URIs, and the Annotation API is in charge of extracting the relevant information from these resources through dereferencing (i.e. extracting the relevant information from the resource identified by the provided URI).

[7] see http://oauth.net/

3 Annotations API

The development of the Annotations API is coordinated by Europeana Foundation (EF) in its API (Scrum [8]) development team, where Europeana Sounds partner AIT is responsible for the development of the Annotations API. The Annotations API is developed to be a part of the Europeana Core REST API[9]. In the first quarter of 2016 the Annotations API will be offered as a public service to external developers.

The specification for the API is constantly under discussion in a working document[10]. When new methods or changes are agreed, they are implemented and deployed to the API. The developer console and the documentation of the released functionality is automatically generated by Swagger [11] (see also Section 3.4.3. Administration). Further documentation on the concrete implementation can be found in Section 3.3 Technical implementation.

The current version of Annotations API can be tested at the following location: http://test-annotations.europeana.eu

A prototype *Annotation Client* which is able to create and read (simple) tags for Europeana objects using the Annotations API can be found here:
http://mediadiscovery.europeana.eu/?action=search&query=&media=true

3.1 Roadmap

The Annotations API will have a major release every six months, with a new minor release after every sprint of development (usually three weeks) in order to quickly iterate and test new features and changes. The overall roadmap for the Annotations API is as follows:

Table 1: Roadmap for the Annotations API

Version	Features	Business goals	Release
1	Simple tag annotations	Support pilot round-trip[12] of annotations between Europeana and Historypin.	July 2015
2	Object linking (Europeana to Europeana) annotations Semantic tagging annotations Indexing/search of annotations Basic moderation capabilities	Open up the Annotations API for user testing to receive more user input. To help make a first annotation use-case available in the Music Collection (most likely to be semantic tagging, using a genre vocabulary).	January 2016
3	Relevancy ranking (for sorting annotations, tags etc.)	Open up the API as a public software, for use by anyone.	July 2016

[8] http://scrummethodology.com/
[9] http://labs.europeana.eu/api/
[10] https://docs.google.com/document/d/1R2_RDuP5E2n0SBnZOqvjedqmKVoBavo6wtS0pN1UJtA/edit#
[11] http://www.swagger.io
[12] Importing annotations from other platforms via "middleware" software which harvests annotations, maps them accordingly and imports them back into Europeana

	Support more annotations: - Figure annotations - Time-based annotations - User sets		
4	Support data providers round-trip (Pundit) Annotations to support: - Rating - Geo-tagging - Metadata correction/completion	Support round-trip of annotations between Europeana Sounds Data Providers and Europeana via Pundit.	January 2017

The current implementation, as of December 2015, is able to create new (simple) tags for Europeana objects, and is able to retrieve annotations by a Europeana record ID, allowing for the utilisation of the full flow of annotations: creating, reading, updating & deleting.

Release 3 and 4 are not entirely defined and still open for refinement, as more input from users will provide better direction for the future roadmap of the Annotations API.

3.2 Designing the Annotations API

A lot of thinking is invested into the design of the Annotations API, as there are often many possible interpretations to the standards and ways to solve the various challenges. In this section the most fundamental of these decisions are explained in detail.

3.2.1 The data model

We adopted the Web Annotation Data Model [13] (or simply WA) as the model and format for exchanging annotations between the client applications and the Annotation API, but also as the way to share with external platforms. The WA is a recent W3C proposal, based on the work of the Open Annotation Community, in particular, the Open Annotation Data Model[14] . This was the model we had chosen and mentioned in earlier deliverables, D2.2 [Ref1] and D2.4 [Ref 3]. The move to this new model is based on the fact that it is likely to become a W3C recommendation. Being a recent specification, it is expected that some changes may happen during the lifetime of the project (as the version from 15th October has showed us).

The development team makes efforts to adopt the changes from new versions of data model as soon as possible (i.e. the ambition is that the major releases comply with the latest version of the standard).

We will now describe the general structure of the model[15]. In WA (as like OA), an annotation is essentially a reified [16] relation between two or more resources, typically a body and target, and conveys

[13] http://www.w3.org/TR/annotation-model/
[14] http://www.openannotation.org/spec/core/
[15] For a more detailed description, please see our master document on modelling: https://docs.google.com/document/d/1I5pFY3WqLn83_mWXxbXGt9_eT3-VRcfkygZ-Ag-k2Io

that the body is related to the (i.e. about a) target. Being reified as a class (oa:Annotation) enables an annotation to be further described, with motivation and provenance information, but also other specific metadata (e.g. a link to the webpage where the annotation can be viewed). An annotation is thus described with the following properties:

- A *URI* which identifies the annotation;
- One or more *target relations* (**oa:hasTarget**) representing a resource or just a part of it that is being annotated;
- One or more *body relations* (**oa:hasBody**) conveying what is intended to be said about the target. A body can also be absent to describe situations where a target is simply bookmarked (highlighted);
- A *motivation* (**oa:hasMotivation**) which expresses the reason why the annotation was created (see list [17] of available motivations);
- A *relation* (**oa:annotatedBy**) to an user agent (foaf:Person) of a client application that created the annotation;
- A *date of creation* (**oa:annotatedAt**) for the annotation should reflect the exact time (expressed in ms) that the annotation was initially created;
- A *relation* (**oa:serializedBy**) to a *software agent* (**prov:SoftwareAgent**) typically a client application (i.e. HistoryPin, Pundit) used to create the annotation;
- A *date of serialization* (**oa:serializedAt**) which captures the date that the annotation was initially stored in the annotation server.

[16] see https://en.wikipedia.org/wiki/Reification_%28computer_science%29
[17] http://www.w3.org/TR/annotation-model/#motivations

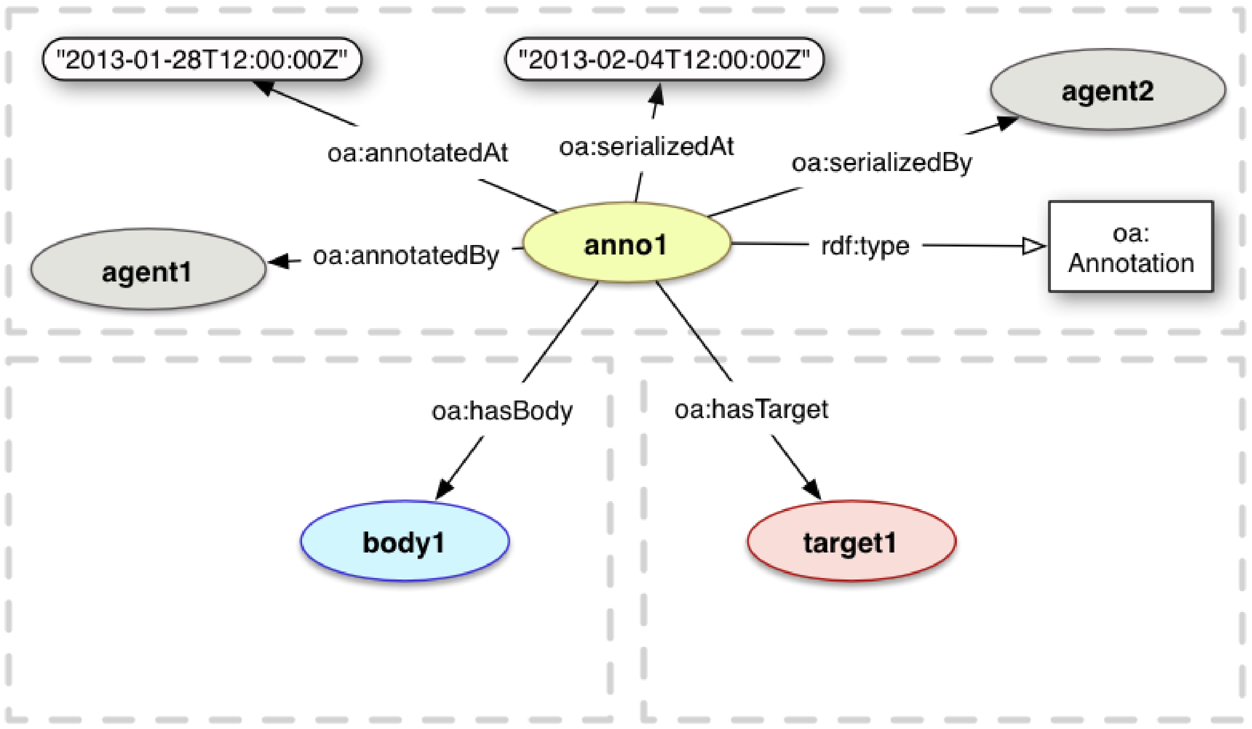


Figure 3: Overview of an Annotation modelled according to the WA specification

With regards to format, the WA is defined using the RDF model, thus making it possible to be serialized in any RDF format. The specification recommends the use of JSON-LD as preferred serialization format and use the Context [18], which was also used to display the example below.

Example X: A simple comment, i.e. an Annotation, made on the HistoryPin.org platform for a musical instrument object (Buccin Trombone), which was provided to Europeana:

```
{
"@context": <Context definition>
"@type": "oa:Annotation",
"@id": "http://data.europeana.eu/annotation/local/1",
"annotatedAt": "2012-11-10T09:08:07Z",
"annotatedBy": "https://www.historypin.org/user/55376",
"serializedAt": "2012-11-10T09:08:07Z",
"serializedBy": "http://data.europeana.eu/provider/Historypin",
"motivation": "oa:tagging",
"body": "A bell with one coil, angled to face forwards",
"target": "http://data.europeana.eu/item/09102/_UEDIN_214"
}
```

Example X: Simple comment formatted in JSON-LD

As can be observed in the description and example above, the WA model offers a quite flexible structure for modelling annotations, in particular for targets and bodies. Given this flexibility, we found the need to define some patterns to promote consistency when exchanging annotations. Concretely, we aim at using a common representation for the annotations defined within different application scenarios

[18] http://www.w3.org/TR/annotation-model/#json-ld-context

[19](e.g. Tunepal, Pundit). Furthermore, we hope to contribute back to both WA and OA communities as best practices at the end of the project. In this regard, we have presented our approach[20] at the Semantic Web in Libraries [21] conference held in Hamburg. We showcased the application scenarios and sharing proposals for modelling that were received with great interest in the community.

3.2.2 Web annotation protocol

As of July 2nd 2015, the W3C Web Annotation Working Group [22] released the specification for the Web Annotation Protocol [23] (or simply WAP) that describes a series of transport mechanisms for creating, managing, retrieving and searching for annotations. The first and current version focuses on the methods to create, retrieve, update, and delete annotations. The remaining methods are expected to be defined soon. It follows other W3C specifications such as the Linked Data Platform [24] (LDP) and current HTTP best practices (e.g. Web Linking [25], Entity Tags [26]). The Annotations API has adopted and implemented the WAP specification (see Section 3.4).

3.2.3 Identification of annotations

Similar to Europeana records, each annotation has its own identifier. Originating from the Round-trip use-case, a requirement was identified to allow an easy mapping between the local identifiers of certain annotation providers and the Europeana resource (i.e. annotation) identifiers.

For this reason, identifiers of annotations in the round-trip scenario will consist of two types of information and following the pattern shown below:

- An identifier of the organisation or application creating the annotation (“provider”).
- A local identifier of the annotation in that organisation or application, or a sequential number (e.g. “123”).

Pattern:

```
http://data.europeana.eu/annotation/{provider}/{index}
```

In the case of Historypin, annotations are stored with identifiers such as:

```
http://data.europeana.eu/annotation/historypin/123
```

[19] See https://docs.google.com/document/d/1Yw1uJdf76v3StXST8x16TReB8FmOLw5LuWOzZz4lSiM
[20] See abstract at http://swib.org/swib15/programme.html#abs18, full presentation at http://www.slideshare.net/antoineisaac/modelling-and-exchanging-annotations-swib15
[21] http://swib.org/swib15/
[22] http://www.w3.org/annotation/
[23] http://www.w3.org/TR/annotation-protocol/
[24] http://www.w3.org/TR/ldp/
[25] https://tools.ietf.org/html/rfc5988
[26] https://tools.ietf.org/html/rfc7230

Whereas the first part ("historypin") denotes the provider and the second part ("123") denotes the local identifier of the annotation used in Historypin. A similar pattern is also used for the metadata records that are provided to Europeana, with the slight difference that the provider part appears as the dataset.

The provider name will be alphanumeric and must be unique across the Europeana API. The provider name will be designated when an API user requests write access to the Annotations API.

To support this use-case, as well as the use-case of clients that do not generate local identifiers, an API client will have to make a decision on the type of identifiers that will be supported by the Annotations API:

a. The API client is a "provider" (Round-trip scenario) such as Historypin and provides its own local identifiers. This means that new annotations will have an identifier which concatenates the provider name and local identifier. The local identifier must be alphanumeric.

b. The API client does not want to nor needs to provide local identifiers. In this case, the Annotations API will generate an identifier based on the provider name and a sequential number.

3.2.4 Moderation principles

Below we present the moderation principles for the Europeana Sounds crowdsourcing infrastructure. These provide the guiding principles for the development of moderation functionality:

1. User Generated Annotations are always stored separately from the original metadata records in the Annotations API. There they are related to the metadata object, as harvested by Europeana. This means enrichments resulting from crowdsourcing will never automatically alter the original metadata record as provided by the Data Provider.

2. End-users of the various crowdsourcing platforms connected to the crowdsourcing infrastructure can only create new annotations, or comment on annotations from other end-users. These new annotations are then stored in the Annotations API, in accordance with the first principle. Comments on annotations by others end-users are related to the original annotations in the Annotation API. End-users can select to make their annotations public, or keep them private. Editing or deleting annotations by other end-users is not possible as an end-user.

3. Instead of supporting end-users with the editing or deleting of annotations by other end-users, we support the evaluation (e.g. ‘flagging’ and ‘liking’) of annotations from other end-users. This aims to encourage crowd-moderation among the end-users. This will support users with the possibility to flag potentially offensive, libellous or spam enrichments. On the other hand it also aims at allowing users to express their explicit support for an already existing enrichment. The exact labels for and types of evaluation will result from user testing. Part of this user research will also be the question whether end-users want to add an evaluation to their own tags, for instance to express a level of certainty about their own annotation. Similarly to the comments mentioned above, these evaluations will be related to the original annotations in the Annotations API.

4. The target of semantic enrichments is restricted to resources from trusted repositories, in order to counter the spamming of links.

5. Utilization of the annotations that can be retrieved from the Annotations API is left up to the policy of the respective data re-users. For instance, these parties can choose to only display annotations that have been evaluated favourably by other users for a specific number of times or never display annotations that have been flagged as part of the crowd-moderation process.

6. Only administrators of the Annotations API can edit/delete annotations. This is meant to be a last resort, for instance to remove annotations of a discriminatory or defamatory nature or to delete spam. Flagged annotations, resulting from the crowd-moderation process mentioned above is one type of annotation that administrators will possibly select for deletion after review.

3.3 Technical implementation

The technical implementation of the Annotation API is performed by following an agile development process, with AIT developers taking part in the regular Scrum calls of the Europeana Development Team. The core development team is closely collaborating with the other project partners for collecting requirements from all stakeholders, writing specifications, standardizing interfaces and integrating the functionality of different tools within the overall crowdsourcing infrastructure, as presented in Section 2.1. *Crowdsourcing Infrastructure Setup*.

3.3.1 Development and testing environment

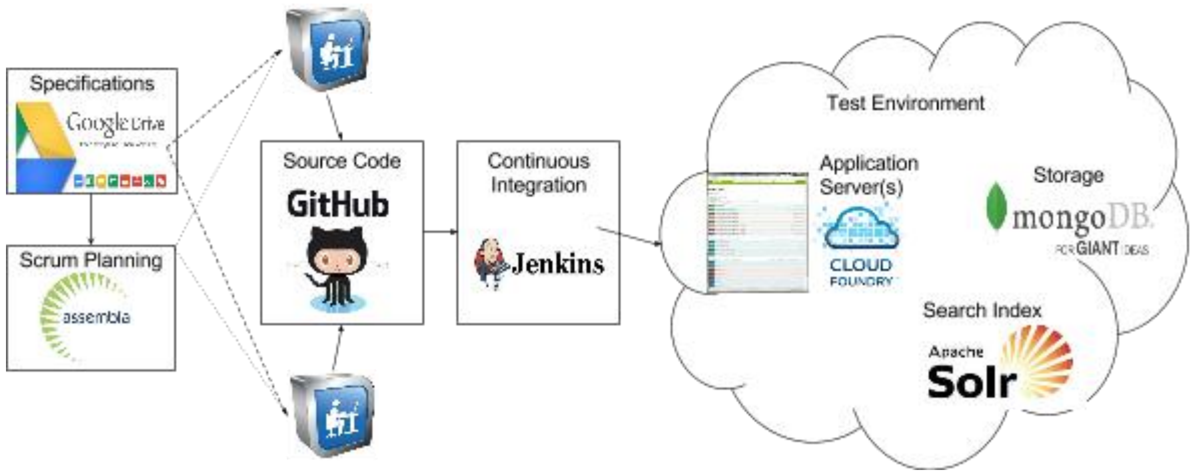


Figure 4: Development Tools and Test Environment

AIT and Europeana are collaborating on specifying and developing the Annotation API. The remote collaboration, the functional and technical requirements regarding to the standardization, openness, quality and scalability of the infrastructure together require an agile development process and appropriate tools for managing planned activities and creation of artefacts. Figure 4 is a sketch of the overall process and the tools used in each step:

- The requirements and specifications of the functionality implemented by the API are collected using *Google Drive* [27] to support a real-time and collaborative environment for editors.

[27] http://drive.google.com

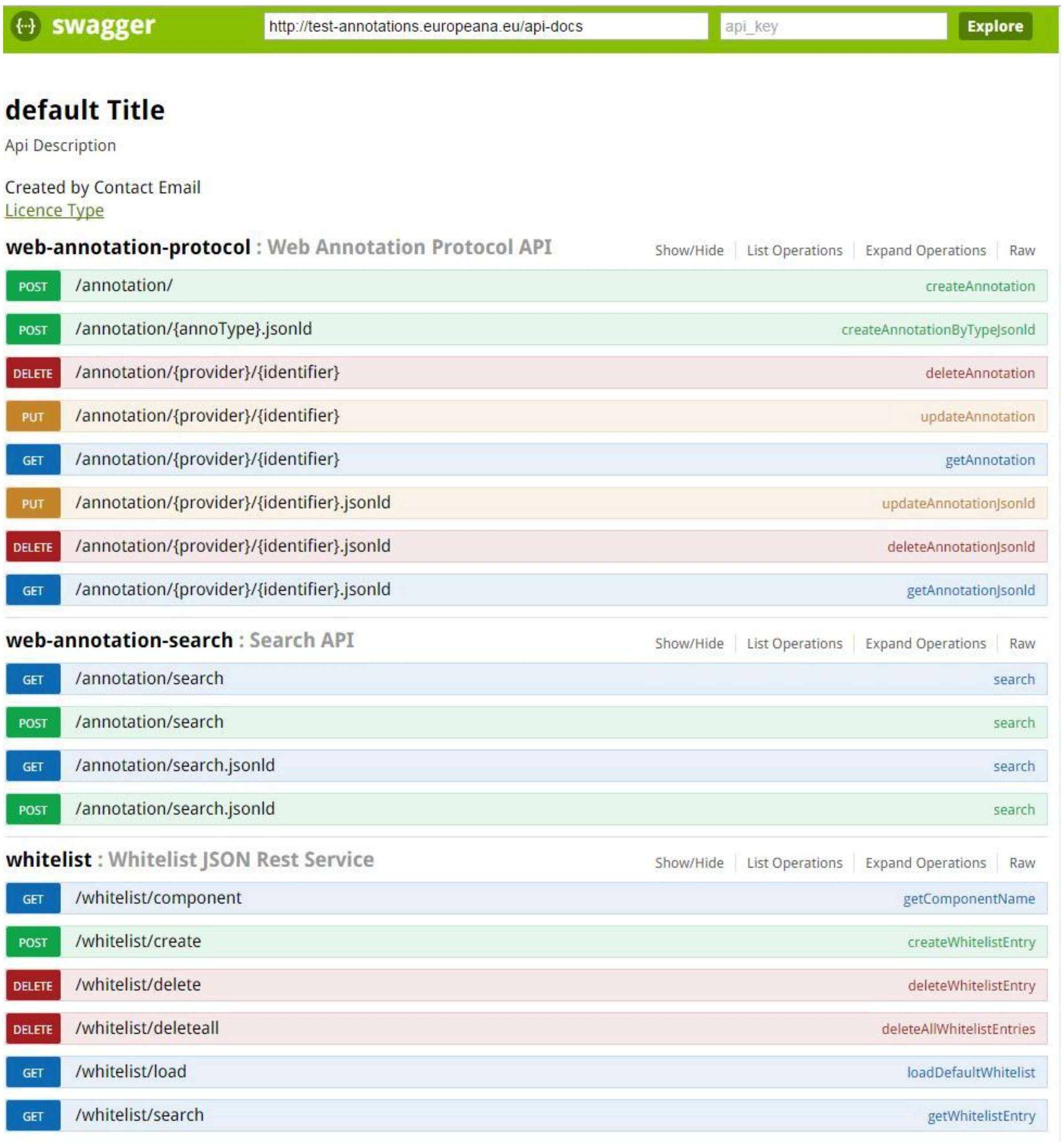
- *Assembla* [28] is used for managing all related activities (i.e. including specification, development, testing, etc.), by following the Scrum model.
- The generated source code is open source and publicly available in *Github* [29].
- *Jenkins* [30] tools are used for building the software artefacts and deploying them on the Test Environment.
- The *Test Environment* uses a cloud based setup, similar to the one used by the Europeana API, including:
  - cloud-based application servers using *Cloud Foundry* [31] infrastructure
  - scalable storage managed by the cloud infrastructure based on *MongoDb* [32] technology
  - search index based on *SolrCloud* [33] technology

3.4 Functional overview

The Annotation API comes with a developer console which serves as a testing and debugging environment for developers. The console is built automatically using the Swagger technology and integrates the technical documentation, describing how to use the individual methods that are supported by the API. The following sections present the current status of development, by using screenshots from the test server accessible at: http://test-annotations.europeana.eu/docs/

The functionality provided by the current version of Annotation API can be grouped in three main categories: a) *managing annotation lifecycle*; b) *searching annotations*; and c) *administration*. As can be observed in Figure 5, this functionality is grouped accordingly in the console by using developer-friendly section names. The management of annotations is implemented by following the specifications from the *Web Annotation Protocol* (see Section 3.2.2), offering specific HTTP endpoints for creating, retrieving, updating and deleting annotations. Advanced Search functionality, following the same design as the Europeana Search API, is available using both GET and POST methods, so that developers can choose the one that is most convenient for them. At the moment, the administrative functionality only includes management of a Whitelist to be used for validating URLs provided in the annotation. A fourth group will be designed and implemented in the next versions of the API to support the moderation process (see Section 3.2.3. Moderation principles).

[28] https://europeanadev.assembla.com/spaces/europeana-npc/
[29] https://github.com/europeana/annotation/
[30] http://jenkins.eanadev.org/job/annotation-api-snapshots/
[31] https://www.cloudfoundry.org/
[32] https://www.mongodb.com
[33] http://lucene.apache.org/solr/



default Title

Api Description

Created by Contact Email
Licence Type

web-annotation-protocol : Web Annotation Protocol API — Show/Hide | List Operations | Expand Operations | Raw

POST	/annotation/	createAnnotation
POST	/annotation/{annoType}.jsonld	createAnnotationByTypeJsonld
DELETE	/annotation/{provider}/{identifier}	deleteAnnotation
PUT	/annotation/{provider}/{identifier}	updateAnnotation
GET	/annotation/{provider}/{identifier}	getAnnotation
PUT	/annotation/{provider}/{identifier}.jsonld	updateAnnotationJsonld
DELETE	/annotation/{provider}/{identifier}.jsonld	deleteAnnotationJsonld
GET	/annotation/{provider}/{identifier}.jsonld	getAnnotationJsonld

web-annotation-search : Search API — Show/Hide | List Operations | Expand Operations | Raw

GET	/annotation/search	search
POST	/annotation/search	search
GET	/annotation/search.jsonld	search
POST	/annotation/search.jsonld	search

whitelist : Whitelist JSON Rest Service — Show/Hide | List Operations | Expand Operations | Raw

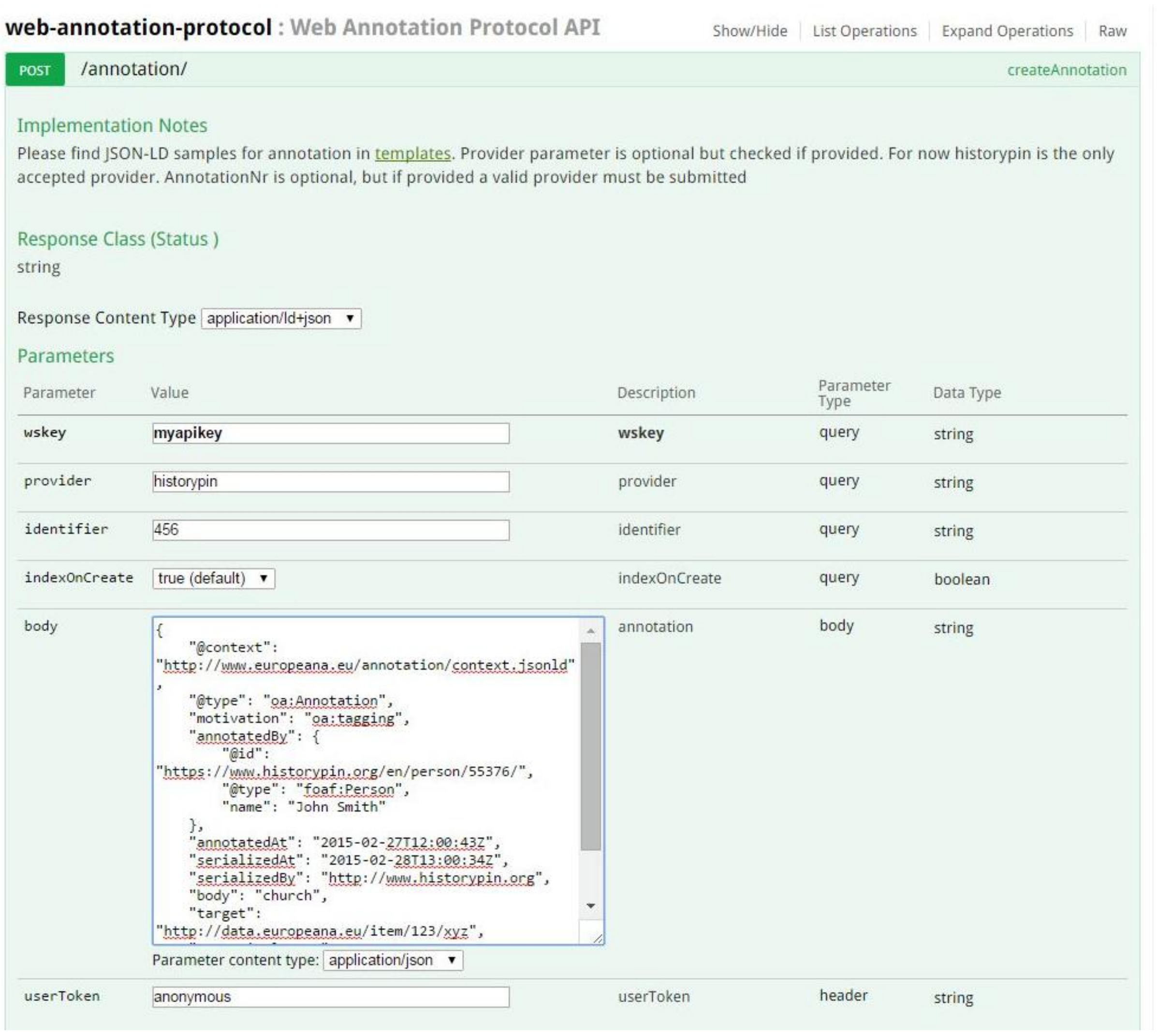
GET	/whitelist/component	getComponentName
POST	/whitelist/create	createWhitelistEntry
DELETE	/whitelist/delete	deleteWhitelistEntry
DELETE	/whitelist/deleteall	deleteAllWhitelistEntries
GET	/whitelist/load	loadDefaultWhitelist
GET	/whitelist/search	getWhitelistEntry

Figure 5: Overview of Annotation API in Developer Console (Swagger)

3.4.1 Create, retrieve, update and delete an annotation

The current version of the Web Annotation Protocol defines the methods for managing annotations (see Section 3.2.2. Web Annotation Protocol). Part of the specification is the format for HTTP requests and corresponding responses. The specific concerns include content negotiation, error handling and appropriate usage of HTTP protocol and status codes. In the following Figures 6, 7, 8 and 9, the console used for creating a new annotation is presented. When building the request the user is allowed to specify the following request parameters:

- *Response content type*, indicating in which format the annotation should be serialized in the http response
- *wskey*, an application key used to identify the annotation client that performs the method invocation
- *provider* indicating the organization that issued the current request
- *identifier* representing the id generated by the provider, for certain use cases this is reused when generating annotation IDs (i.e. URIs)
- *body* represents the payload of the HTTP request which carries the annotation represented in Web Annotation data model and serialized using json-ld format
- *userToken* represents session scoped identifier for the user creating the annotation
- *annoType* represents a convenient way for ensuring that the appropriate type of annotation is created. This is closely mapped to the standard motivation field.



web-annotation-protocol : Web Annotation Protocol API

/annotation/

Implementation Notes

Please find JSON-LD samples for annotation in templates. Provider parameter is optional but checked if provided. For now historypin is the only accepted provider. AnnotationNr is optional, but if provided a valid provider must be submitted

Response Class (Status)

string

Response Content Type application/ld+json

Parameters

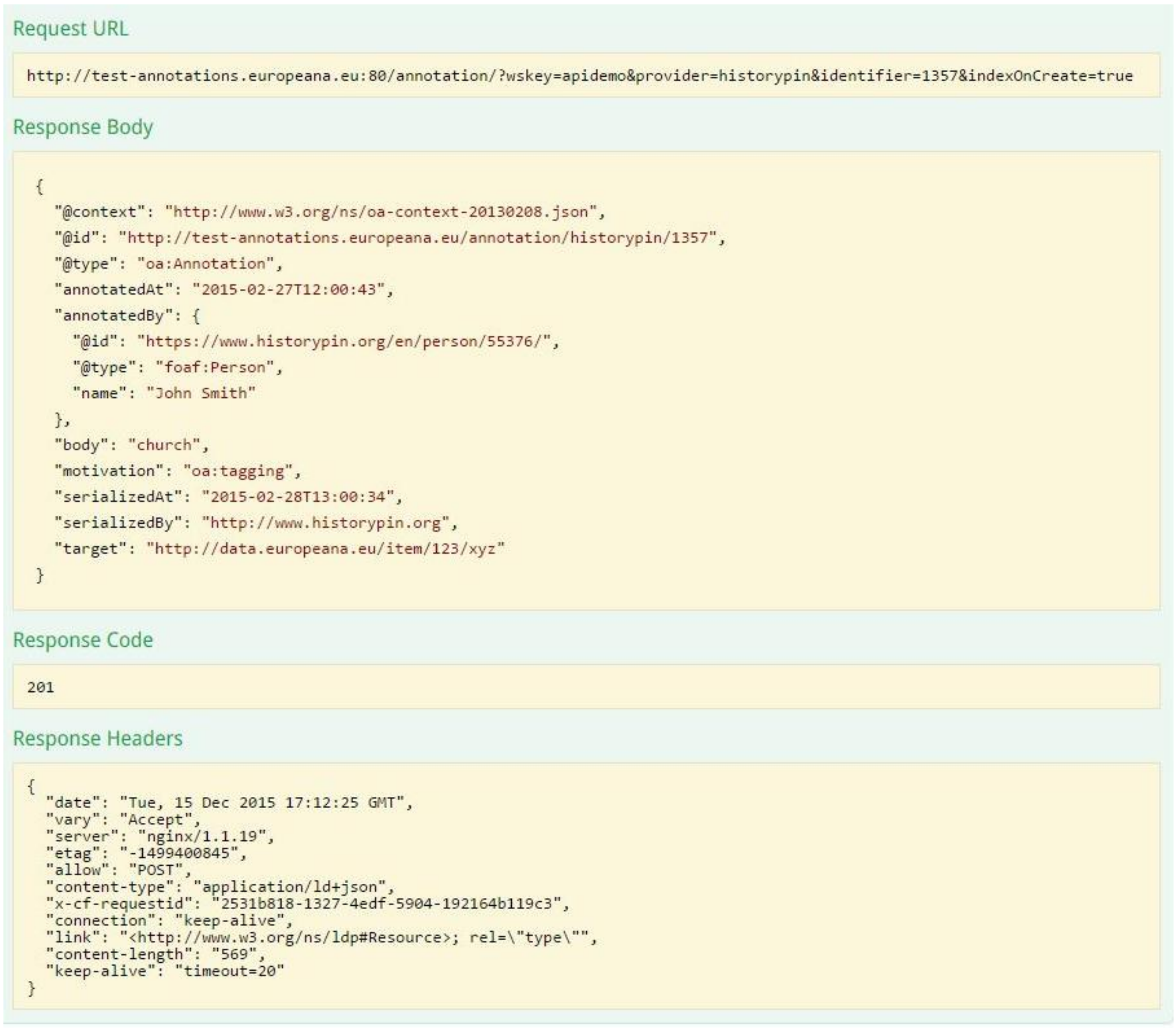
Parameter	Value	Description	Parameter Type	Data Type
wskey	myapikey	wskey	query	string
provider	historypin	provider	query	string
identifier	456	identifier	query	string
indexOnCreate	true (default)	indexOnCreate	query	boolean
body	(see below)	annotation	body	string
userToken	anonymous	userToken	header	string

```
{
    "@context":
"http://www.europeana.eu/annotation/context.jsonld"
,
    "@type": "oa:Annotation",
    "motivation": "oa:tagging",
    "annotatedBy": {
        "@id":
"https://www.historypin.org/en/person/55376/",
        "@type": "foaf:Person",
        "name": "John Smith"
    },
    "annotatedAt": "2015-02-27T12:00:43Z",
    "serializedAt": "2015-02-28T13:00:34Z",
    "serializedBy": "http://www.historypin.org",
    "body": "church",
    "target":
"http://data.europeana.eu/item/123/xyz",
```

Parameter content type: application/json

Figure 6: Create annotations interface (Swagger Console)

If the submitted annotation is successfully created, the following response will be shown in the console. This includes the URL used by the request, the response body representing the full annotation object using the requested serialization (i.e. JSON-LD[34]), the appropriate HTTP status code and a set of response headers carrying information useful for method discovery, caching, parsing response payload.


Request URL

```
http://test-annotations.europeana.eu:80/annotation/?wskey=apidemo&provider=historypin&identifier=1357&indexOnCreate=true
```

Response Body

```
{
  "@context": "http://www.w3.org/ns/oa-context-20130208.json",
  "@id": "http://test-annotations.europeana.eu/annotation/historypin/1357",
  "@type": "oa:Annotation",
  "annotatedAt": "2015-02-27T12:00:43",
  "annotatedBy": {
    "@id": "https://www.historypin.org/en/person/55376/",
    "@type": "foaf:Person",
    "name": "John Smith"
  },
  "body": "church",
  "motivation": "oa:tagging",
  "serializedAt": "2015-02-28T13:00:34",
  "serializedBy": "http://www.historypin.org",
  "target": "http://data.europeana.eu/item/123/xyz"
}
```

Response Code

```
201
```

Response Headers

```
{
  "date": "Tue, 15 Dec 2015 17:12:25 GMT",
  "vary": "Accept",
  "server": "nginx/1.1.19",
  "etag": "-1499400845",
  "allow": "POST",
  "content-type": "application/ld+json",
  "x-cf-requestid": "2531b818-1327-4edf-5904-192164b119c3",
  "connection": "keep-alive",
  "link": "<http://www.w3.org/ns/ldp#Resource>; rel=\"type\"",
  "content-length": "569",
  "keep-alive": "timeout=20"
}
```

Figure 7: HTTP response confirming successful creation of Annotation

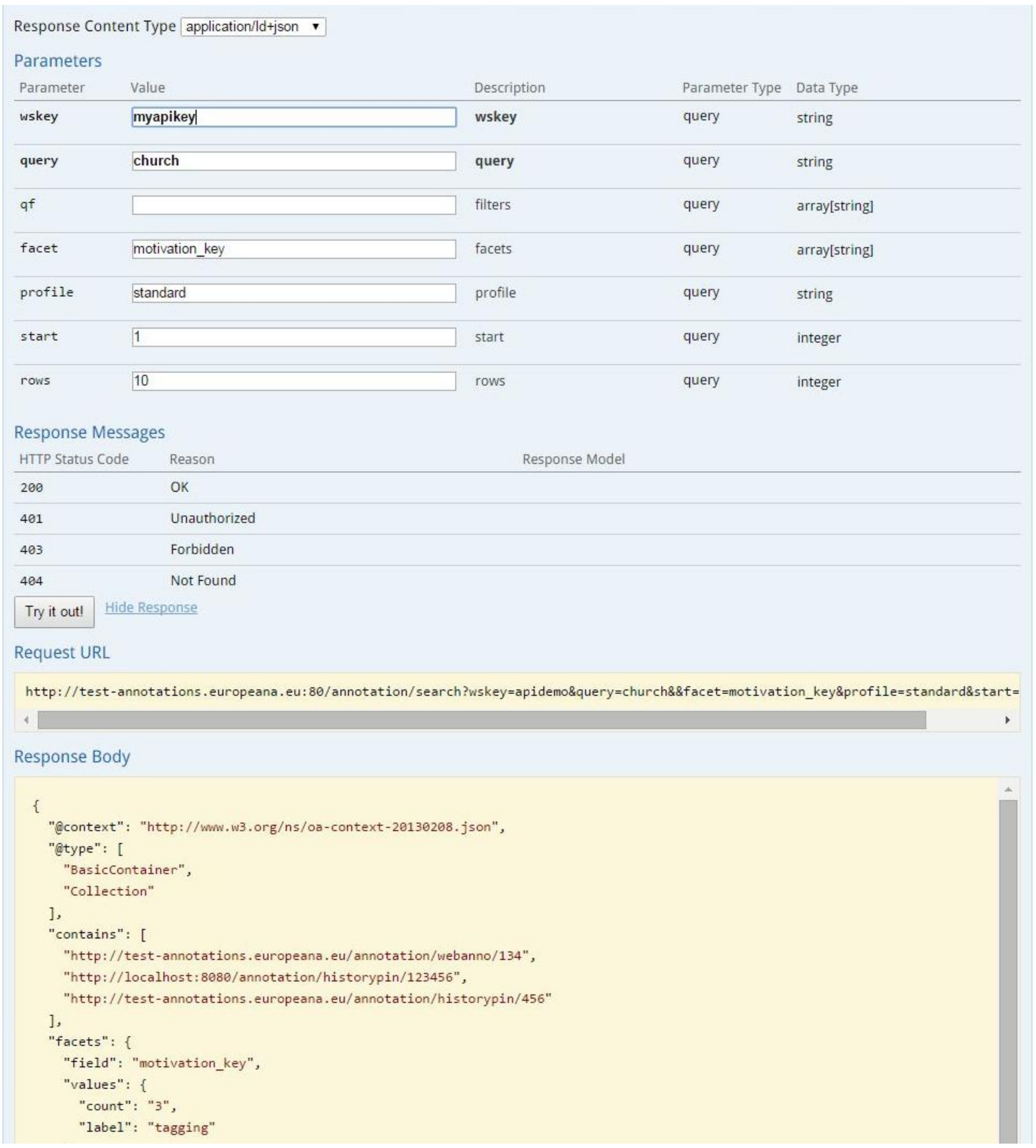
3.4.2 Searching for annotations

The Search API is fundamental for retrieving annotations. It is essential for users to discover annotations (their own, or annotations made by other users), but also for client applications as a means to retrieve and display all the annotations made for a given resource (i.e. a metadata record provided to Europeana or a media file). A typical use case is a user wanting to find all items that were marked with the same tag (e.g. “Christmas Carols”). In order to satisfy various information needs, coming from different use cases, a general search method was defined and implemented.

[34] http://www.w3.org/TR/json-ld/

As the WAP has not defined a method for searching annotations yet, we have chosen to design one ourselves. However, we will keep in touch with the community to see if such method becomes available during the lifetime of the project. This method was designed following the API guidelines currently defined at Europeana and by adopting the same principles as the Europeana Search API, but with some adjustments to be consistent with the methods defined by WAP. It supports requests for free text search, query filters, faceted search and pagination of search results. With regard to responses, and after seeking advice from the WA community, we chose to follow the draft specification using LDP paging [35]. The method is capable of responding with different levels (degrees) of detail as requested through the profile parameter (similar to the Europeana Search API) but also as defined in LDP paging, so that it complies with the specification. At the moment only the *facets* and *minimal* profiles have been implemented, while *standard* and *full* profiles are planned to be implemented in the next releases.

[35] http://w3c.github.io/web-annotation/protocol/wd/paging.html


Response Content Type application/ld+json ▼

Parameters

Parameter	Value	Description	Parameter Type	Data Type
wskey	myapikey	wskey	query	string
query	church	query	query	string
qf		filters	query	array[string]
facet	motivation_key	facets	query	array[string]
profile	standard	profile	query	string
start	1	start	query	integer
rows	10	rows	query	integer

Response Messages

HTTP Status Code	Reason	Response Model
200	OK	
401	Unauthorized	
403	Forbidden	
404	Not Found	

Request URL

```
http://test-annotations.europeana.eu:80/annotation/search?wskey=apidemo&query=church&&facet=motivation_key&profile=standard&start=
```

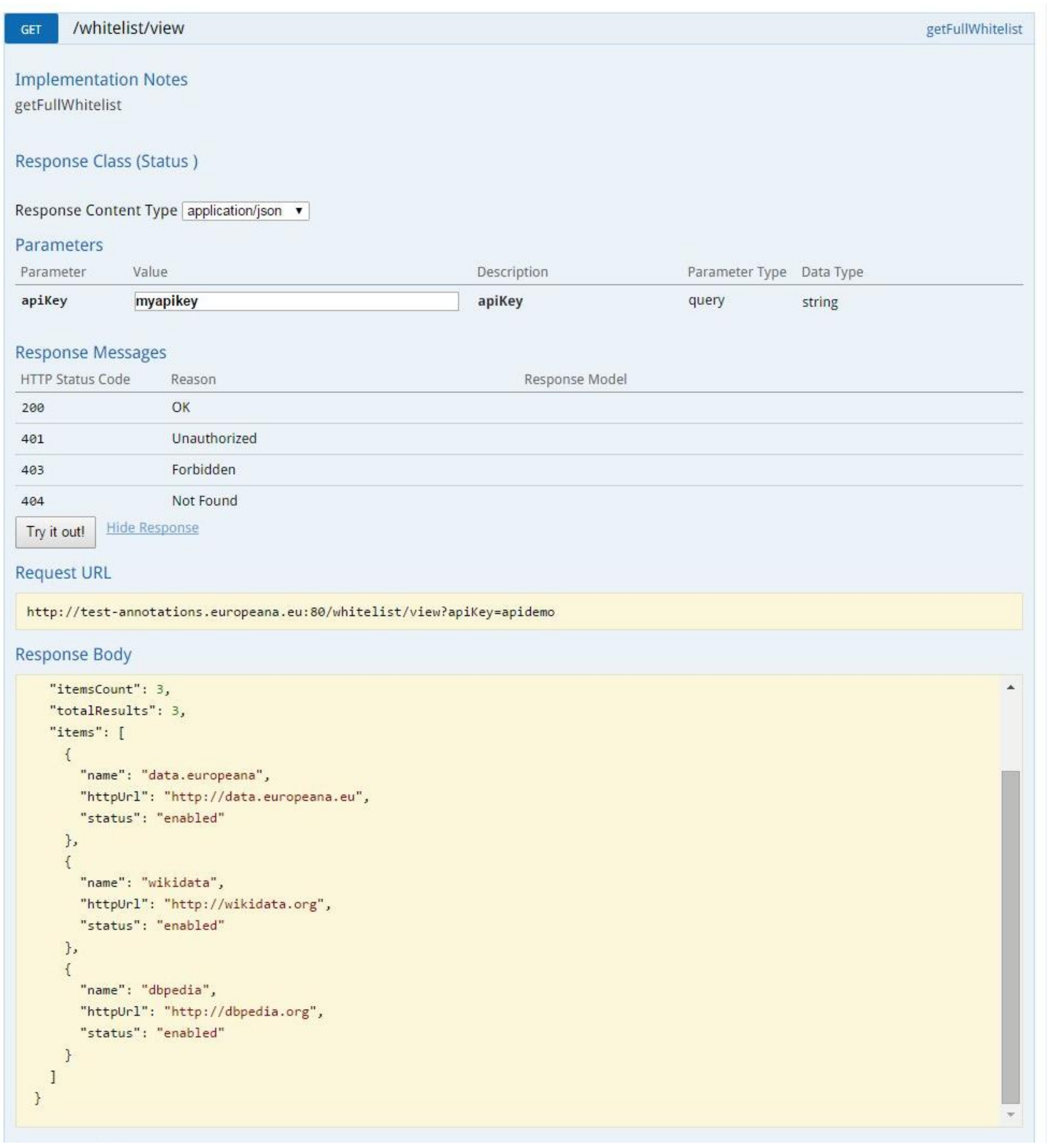
Response Body

```
{
  "@context": "http://www.w3.org/ns/oa-context-20130208.json",
  "@type": [
    "BasicContainer",
    "Collection"
  ],
  "contains": [
    "http://test-annotations.europeana.eu/annotation/webanno/134",
    "http://localhost:8080/annotation/historypin/123456",
    "http://test-annotations.europeana.eu/annotation/historypin/456"
  ],
  "facets": {
    "field": "motivation_key",
    "values": {
      "count": "3",
      "label": "tagging"
```

Figure 8: Search Annotations functionality (Swagger Console)

3.4.3 Administration

Functionality included in the administration group deals with the runtime configuration of the Annotation API. While there is more functionality proposed to be performed exclusively by users having assigned the administrator role, the current version is providing support for management of whitelist entries. These are used to validate bodies and targets, in which the URLs must reference resources coming from trusted repositories. The whitelist API provides functionality for managing the list of trusted repositories at runtime and to search entries by name or URL. The list of predefined trusted repositories can be manually loaded during the system setup, containing the entries displayed in Figure 9 below.



Parameter	Value	Description	Parameter Type	Data Type
apiKey	myapikey	apiKey	query	string

HTTP Status Code	Reason	Response Model
200	OK	
401	Unauthorized	
403	Forbidden	
404	Not Found	

```
http://test-annotations.europeana.eu:80/whitelist/view?apiKey=apidemo
```

```
    "itemsCount": 3,
    "totalResults": 3,
    "items": [
      {
        "name": "data.europeana",
        "httpUrl": "http://data.europeana.eu",
        "status": "enabled"
      },
      {
        "name": "wikidata",
        "httpUrl": "http://wikidata.org",
        "status": "enabled"
      },
      {
        "name": "dbpedia",
        "httpUrl": "http://dbpedia.org",
        "status": "enabled"
      }
    ]
  }
```

Figure 9: Administration functionality (Swagger Console)

3.5 Next steps

Before the Annotations API is able to support its first use-case, to allow users to add genres to music objects in the Europeana Music Collection, a few important things have to be specified and implemented first:

3.5.1 Authentication and authorisation

Any user who wants to create an annotation in the Europeana Music Collection needs to be a MyEuropeana [36] user. As not every API client will initially be allowed to create annotations, as users

[36] http://labs.europeana.eu/api/myeuropeana

must have assigned a given role and permissions. Consequently, the API needs to be aware of user roles, functionality that is not implemented or specified yet. There is also an outstanding challenge concerning the authorisation and authentication in the round-trip scenario. As this is an asynchronous process, there is no ability to use a protocol such as OAuth

3.5.2 Entities

One important aspect of semantic enrichment (e.g. especially for semantic tagging), is that a user interface ideally has to offer a way to the user to search within or browse a list of concepts to choose from to help the user select the correct entity. For this, Europeana is working on implementing the Entity API[37], which will facilitate retrieval and search of entities. User interfaces (such as Europeana Collections) will employ this to assist users when creating semantic tags. Vocabularies have to be defined in order to make this work, while specifications will start by concentrating on the scenario in which users will add music genres to objects in the Europeana Music Collection.

4 Historypin traditional music pilot

This section describes the advancement of the *Crowdsourcing through specialised platforms scenario* (as introduced in D2.2 [Ref 1]), which focusses on the challenge of music identification. It describes the underlying user research, the proposed technical solution, and a timeline for the development and integration.

As part of WP2, Shift is contributing to Task 2.1 - *Crowdsourcing*. As the crowdsourcing in WP2 is focused on micro-tasks that generate genuinely useful additions, Historypin has proposed a traditional music pilot that brings archival sounds holdings to knowledgeable users that might not be aware of these collections by themselves.

4.1 Prototype Tunepal widget

4.1.1 The need

Anecdotal evidence gathered by Historypin through informal conversations with traditional musicians early in 2015 suggested that a service to help traditional musicians identify and learn new tunes through listening to archival recordings would be useful. To understand this better, Historypin undertook user consultation in the form of a survey to answer questions about how traditional musicians hear new tunes, how they identify and find them, and how they subsequently learn these new tunes.

The result of this survey was an insight in the workflow that traditional musicians follow when they are identifying and learning new tunes, which is detailed below. To identify and learn a tune, the musician would:

1. Record the tune at a session or a feis (traditional music festival)

[37] https://docs.google.com/document/d/16Rw_qlSpINxztGpI5sM6NcXhnQeZkW3hRA1IYjFtE1w/edit#heading=h.l2fg46yn5tej

2. Upload it into Tunepal to find a name
3. Use the name to search for recordings of that tune on Google, YouTube or TheSession.org
4. Pick a recording they like best and learn that by ear

Based on this survey among traditional musicians, it is clear that there is a way to use the traditional music archive recordings that are being shared as part of Europeana Sounds to streamline a process that traditional musicians are using in the real world as part of their daily engagement with traditional music.

The traditional music pilot that Historypin has proposed combines steps in the above workflow and eliminate the need for musicians to first find the name of a tune and then use different sources to find high-quality archival recordings of it. As a result, musicians would find it easier to identify canonical tunes that they can learn by ear and to discover and connect to the traditional music archives that hold these tunes.

4.1.2 The solution

As part of our work on Europeana Sounds and to provide real connections between the traditional music scene and music archives, Historypin is working together with the developer of Tunepal to improve the Tunepal app in such a way that it becomes a website-based app, into which you can play a traditional Irish or Scottish tune and which will then surface archival recordings of that tune, pulled from the Europeana database. These tunes are supplied to Europeana by Europeana Sounds data providers Tobar an Dualchais and Comhaltas Ceoltoiri Eireann. The platform is extensible to any Europeana-provided datasets, though at the moment only these two providers offer data at a suitable level of access and description, so the Europeana search query is tailored to just these providers.

Shift believes that such an app will lead to more discovery of archival material related to Irish and Scottish traditional music, and to greater enjoyment among traditional musicians as they can find authoritative sources for tune names and other metadata, as well as different versions of the same tune, in an easier and more engaging way. We want to make the archives more relevant in the context of teaching, learning and performing traditional music.

Between March and August 2015, Historypin has worked closely together with Professor Bryan Duggan, the creator of Tunepal, to link the archival recordings to Tunepal through the Europeana API. These archival recordings were linked by using the existing Tunepal transcription functionality to first transcribe the music and then to identify candidate track names against a corpus of user-contributed tune titles at TheSession.org. Using these track names as a pivot, the Europeana Search API was used to query the provided datasets. In addition, Tunepal has been updated to work in a web browser rather than in a mobile app, and a bespoke interface to display the archival recordings next to the tune names has been created.

4.1.3 Description of widget

This Tunepal app has taken the form of a HTML/CSS/JavaScript widget that:

1. Allows the user to record a 12 second piece of music:

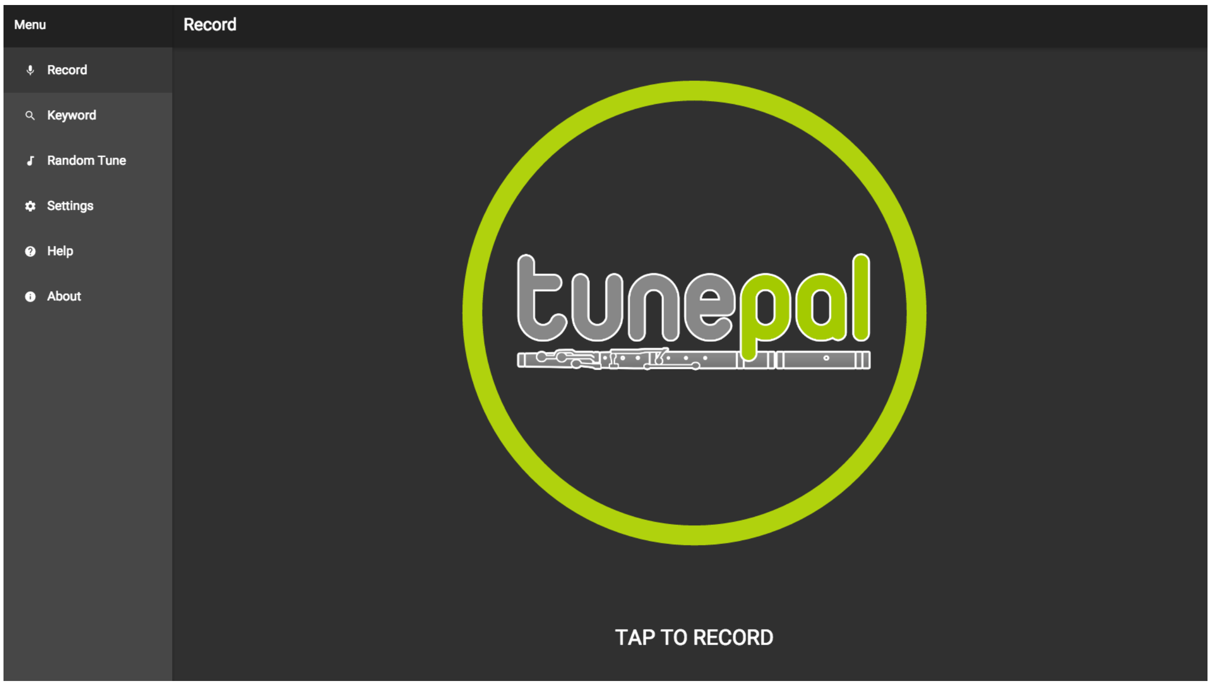
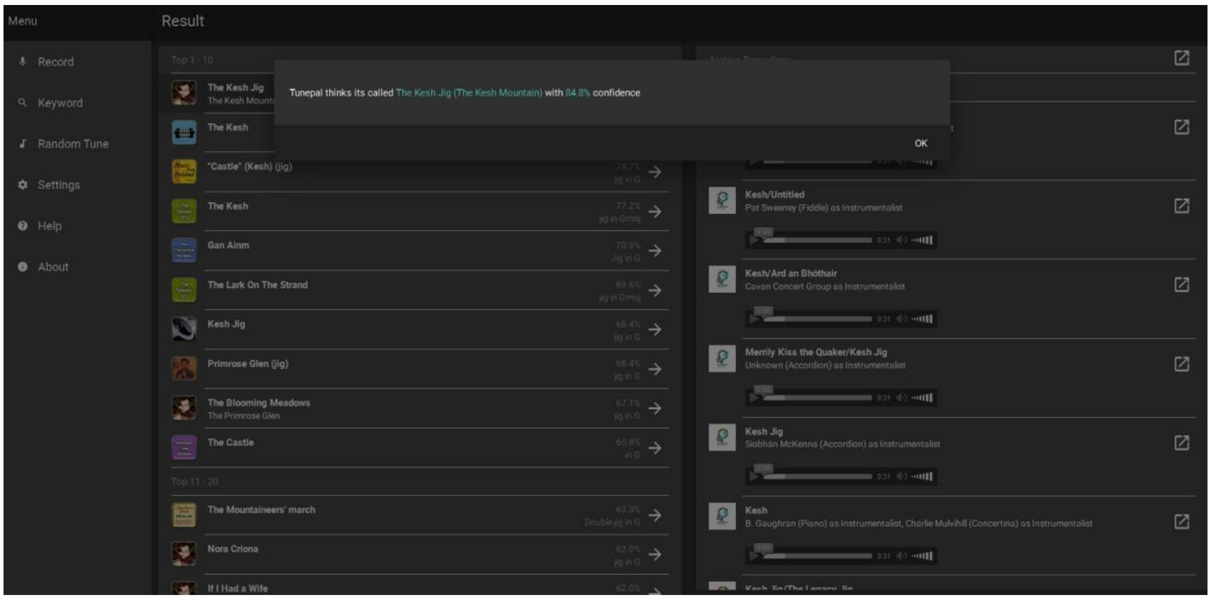


Figure 10: Tunepal homescreen

2. Connects to Tunepal to get some suggested high-probability tune names:



Tunepal thinks its called The Kesh Jig (The Kesh Mountain) with 84.8% confidence

Top 1 - 10

The Kesh Jig
The Kesh Mount

The Kesh

"Castle" (Kesh) (jig) 79.7% jig in G

The Kesh 77.2% jig in Gmaj

Gan Ainm 70.9% Jig in G

The Lark On The Strand 69.6% jig in Gmaj

Kesh Jig 68.4% jig in G

Primrose Glen (jig) 68.4% jig in G

The Blooming Meadows
The Primrose Glen 67.1% jig in G

The Castle 65.8% in G

Top 11 - 20

The Mountaineers' march 63.3% Double jig in G

Nora Criona 62.0% jig in G

If I Had a Wife 62.0%

Kesh/Untitled
Pat Sweeney (Fiddle) as Instrumentalist

Kesh/Ard an Bhóthair
Cavan Concert Group as Instrumentalist

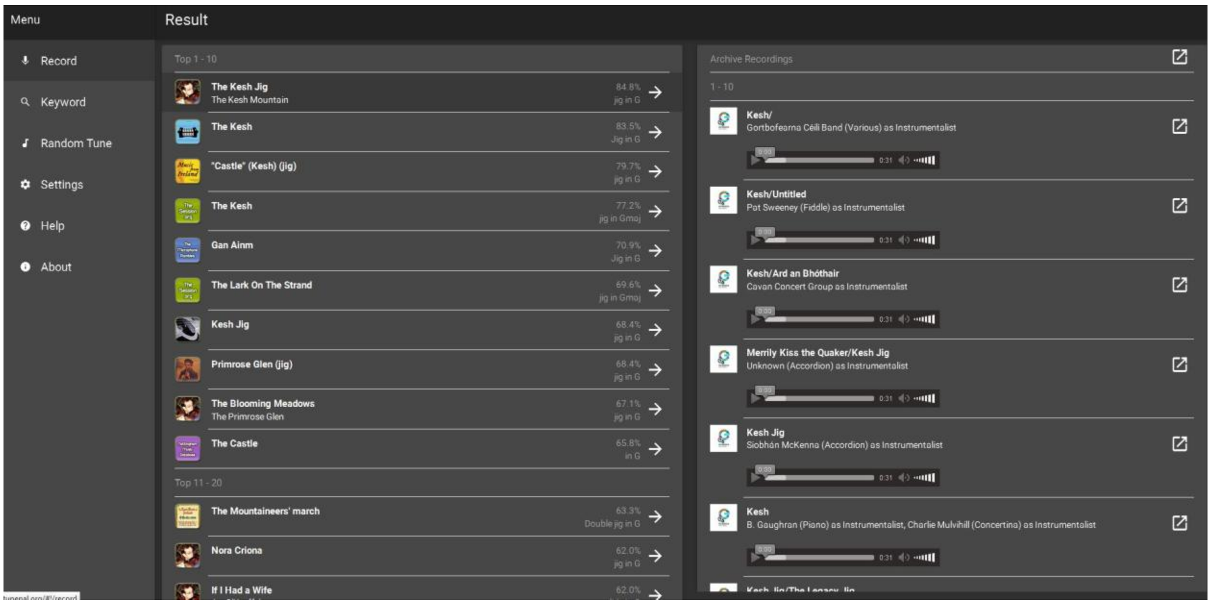
Merrily Kiss the Quaker/Kesh Jig
Unknown (Accordion) as Instrumentalist

Kesh Jig
Siobhán McKenna (Accordion) as Instrumentalist

Kesh
B. Gaughran (Piano) as Instrumentalist, Charlie Mulvihill (Concertina) as Instrumentalist

Figure 11: Tunepal query-by-playing result

3. Queries the Europeana Search API to get back archive recordings that are likely matches:



Result

Top 1 - 10

The Kesh Jig The Kesh Mountain	84.8% jig in G	→
The Kesh	83.5% Jig in G	→
"Castle" (Kesh) (jig)	79.7% jig in G	→
The Kesh	77.2% jig in Gmaj	→
Gan Ainm	70.9% Jig in G	→
The Lark On The Strand	69.6% jig in Gmaj	→
Kesh Jig	68.4% jig in G	→
Primrose Glen (jig)	68.4% jig in G	→
The Blooming Meadows The Primrose Glen	67.1% jig in G	→
The Castle	65.8% in G	→

Top 11 - 20

The Mountaineers' march	63.3% Double jig in G	→
Nora Criona	62.0% jig in G	→
If I Had a Wife	62.0%	→

Archive Recordings

1 - 10

Kesh/
Gortbofearna Céili Band (Various) as Instrumentalist

Kesh/Untitled
Pat Sweeney (Fiddle) as Instrumentalist

Kesh/Ard an Bhóthair
Cavan Concert Group as Instrumentalist

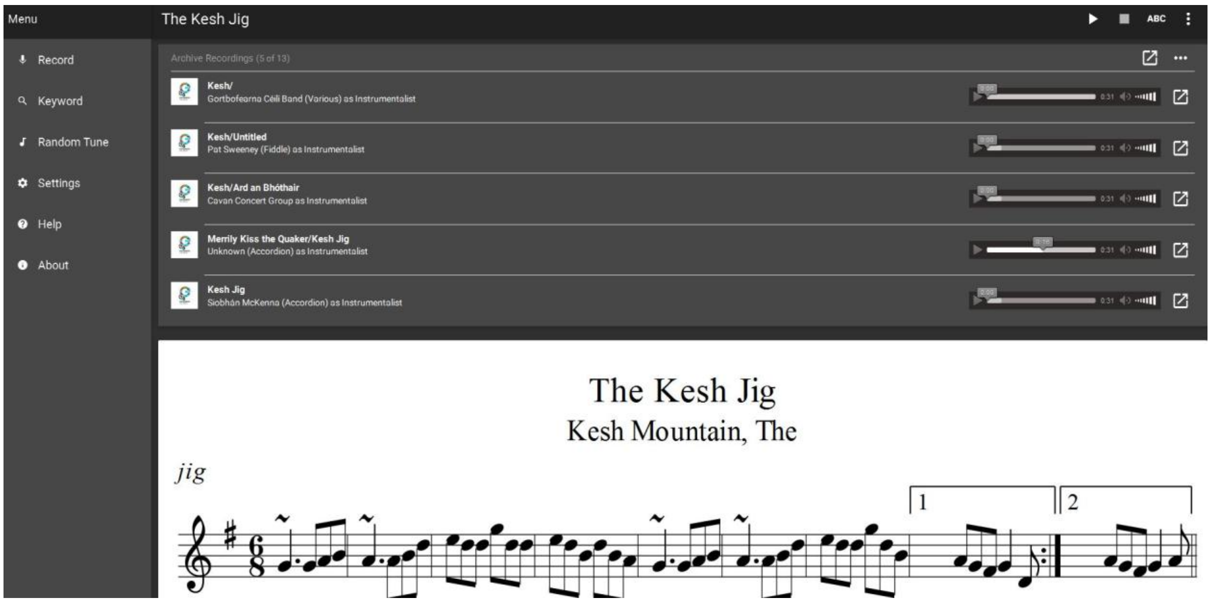
Merrily Kiss the Quaker/Kesh Jig
Unknown (Accordion) as Instrumentalist

Kesh Jig
Siobhán McKenna (Accordion) as Instrumentalist

Kesh
B. Gaughran (Piano) as Instrumentalist, Charlie Mulvihill (Concertina) as Instrumentalist

Figure 12: Tunepal query-by-playing results

4. Lets the user listen to some of those audio candidates:



The Kesh Jig

Archive Recordings (5 of 13)

Kesh/
Gortbofearna Céili Band (Various) as Instrumentalist

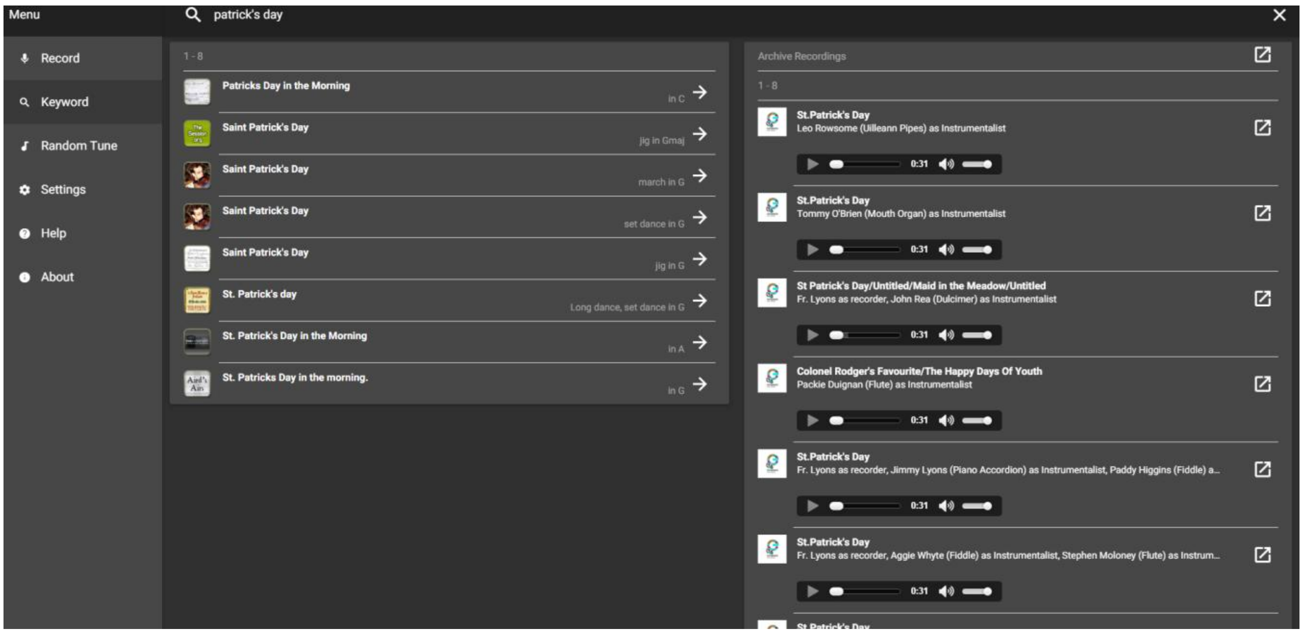
Kesh/Untitled
Pat Sweeney (Fiddle) as Instrumentalist

Kesh/Ard an Bhóthair
Cavan Concert Group as Instrumentalist

Merrily Kiss the Quaker/Kesh Jig
Unknown (Accordion) as Instrumentalist

Kesh Jig
Siobhán McKenna (Accordion) as Instrumentalist

Figure 13: Tunepal individual tune result with archive recordings and staff notation



1 - 8

Patricks Day in the Morning — in C
Saint Patrick's Day — jig in Gmaj
Saint Patrick's Day — march in G
Saint Patrick's Day — set dance in G
Saint Patrick's Day — jig in G
St. Patrick's day — Long dance, set dance in G
St. Patrick's Day in the Morning — in A
St. Patricks Day in the morning. — in G

1 - 8

St.Patrick's Day
Leo Rowsome (Uilleann Pipes) as Instrumentalist
0:31

St.Patrick's Day
Tommy O'Brien (Mouth Organ) as Instrumentalist
0:31

St Patrick's Day/Untitled/Maid in the Meadow/Untitled
Fr. Lyons as recorder, John Rea (Dulcimer) as Instrumentalist
0:31

Colonel Rodger's Favourite/The Happy Days Of Youth
Packie Duignan (Flute) as Instrumentalist
0:31

St.Patrick's Day
Fr. Lyons as recorder, Jimmy Lyons (Piano Accordion) as Instrumentalist, Paddy Higgins (Fiddle) a...
0:31

St.Patrick's Day
Fr. Lyons as recorder, Aggie Whyte (Fiddle) as Instrumentalist, Stephen Moloney (Flute) as Instrum...
0:31

St Patrick's Day

Figure 14: Tunepal search result

5. Records (as an enrichment) the link between input and matched recording, if the user indicates that there is a match.

The latest version of Tunepal which incorporates steps 1-4 is available at: www.tunepal.org

A presentation by Bryan Duggan on Tunepal and a live demonstration of how Tunepal works can be found at 44.00 minutes in on this video: https://vimeo.com/143878288

4.2 Testing at the Fleadh

After Tunepal had been updated to include the above specifications, Historypin tested the first version of this app at the Fleadh Cheoil [38] in Sligo, the largest Irish traditional music festival in the world. Historypin tested at the Scoil Eigse [39], which is an Irish music masterclass organised as part of the Fleadh.

In total 40 users, ranging roughly from 10 years to 65 years, tested the prototype. Tunepal was displayed on a tablet and the users were first asked whether they had used Tunepal before, and what they thought it could do for them.

The users were then asked to play a tune into the device and to explore the content that was returned to them. The interviewer recorded which features the user found and used correctly, whether the user enjoyed using Tunepal and what they found particularly difficult.

[38] http://www.irishtimes.com/culture/music/fleadh-cheoil-biggest-irish-trad-music-festival-launched-1.2311635
[39] https://comhaltas.ie/blog/post/scoil_eigse_2015/

Figure 15: User testing with students of traditional music at the Fleadh

Some research results are listed below:

- 17 users reported having used a version of Tunepal before, 23 users had never used it before
- the browser-based version of Tunepal (which was being tested here) was less accurate than the native app version. This seems to be related to several factors, the amount of memory available to the browser on a mobile device and the sample rate of the microphone which is available to the application. These are being addressed in further refinement of the transcription algorithm.
- rhythmic consistency is very important for the app to be able to recognise tunes accurately, which means the app was not particularly accurate for younger, less experienced players
- based on anecdotal evidence, this version of Tunepal is very helpful for experts (such as tutors in traditional music) as well as beginners, especially because the archival recordings are being surfaced
- the 30-second archival tracks that have been provided by Comhaltas through Europeana (as a replacement for the full-length tracks) are not long enough, as most recordings are of sets and the queried tune might be the second or third tune in the set. Historypin and Tunepal created the interface based on the accessibility of the full length archival tracks. This has been raised with the data provider.

- the Tunepal interface needs more prompts and more labelling, to make it easier for the user to understand. For example, the function of the arrow in the tune results page is very unclear to users
- the 'extra' menu isn't easily discoverable by users and does not add any value for the particular segment of users that we have tested with
- at the moment, the interface does not offer users a quick and obvious way to test whether the tune they have played and the results Tunepal is giving are a match

Figure 16: User testing with students of traditional music at the Fleadh in Sligo, Ireland

4.3 Historypin API development

The Europeana Sounds project has set a goal of "support[ing] discovery and use by improving metadata through innovative methods including semantic enrichment and crowdsourcing", which will be met, in part, through a new round-trip workflow of metadata with the crowdsourcing platform Historypin.

The historypin.org platform is being extended technically in two key areas to support these goals:

- Historypin will now have the capability of sending user-contributed crowdsourced enrichments (in the form of WA annotations encoded in JSON-LD, see Section X) back to the Annotations API , either through a push protocol to the Annotations API or via a pull initiated by Europeana from a new Historypin Annotations API service. This annotations round-trip extension [40] has been documented in the various other technical deliverables of WP2.

[40] https://docs.google.com/document/d/1eP6CVFLtKqSCduhcuYnqPEKQR7_Za0cizNYE0SQ-6Og

- The Historypin Search API service will be extended to allow for external repositories to query, in a simple JSON-based format, all data objects held within the Historypin repository, using a JSON-based extension of Dublin Core metadata fields, and to perform searches on the repository to identify Europeana-originated or user-annotated records.

This round-trip of annotations data is being developed in project Tasks T2.1.2 and T2.2 and will be used in the Tunepal exploration pilot in T2.4, under the supervision of the project's Technical Coordination Group.

The project has a further goal of "Put[ting] in place policies (and in connection with WP5) infrastructural preconditions allowing enrichments to be re-ingested in the information systems of the contributing archives, wherever relevant." [41] This demonstration of value to the contributing data providers is a key strategic goal of Europeana and of the project. Meeting this goal requires that enrichments or contextualised additions to partner metadata occurring on another platform be available for use. This is strategic driver for every API-driven use of Europeana metadata, and is also true of the Historypin crowdsourcing and contribution platform.

The historypin.org platform at the beginning of the project was self-contained — it did not have a way of bringing information into the repository in an automated way, nor a way to make its data available via an API. The work in the project will make the platform interoperable and responsive to the needs of end-users of the Sounds data contributed in the project.

In addition, the platform did not expose annotations or enrichments in any standard format, and except for the historypin.org website itself there was no way for these user-contributed links and data enrichments to be used. As part of the technical work on the Europeana Sounds project, the platform will become able to produce annotations compliant with the WA, and to make its data available in machine-readable formats for other uses, including by the original data providers.

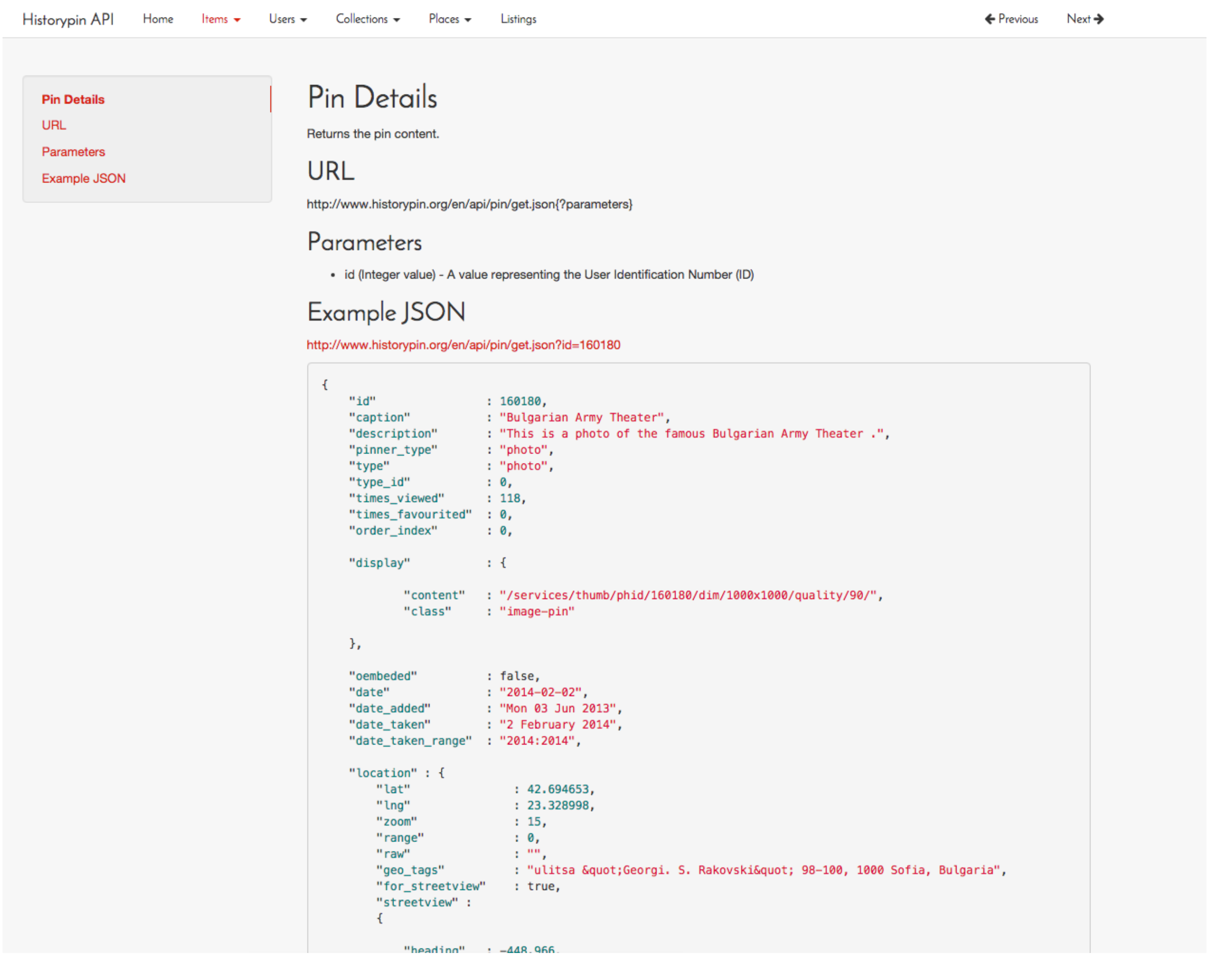
Historypin will play a key role as one of the first data repositories to offer a two-way round trip of metadata with Europeana. The historypin.org platform will collect metadata relating to user-contributed enrichments in the form of keyword (subject) tags, responses (comments) and contextual links between items within the Europeana repository, and then have the capability of contributing this to the Europeana repository.

The first user-facing application using this technology will be the Tunepal based find-by-playing query service described in the next section of this document, in which users play music or load a contemporary recording into an application which then matches these music objects against archival recordings contributed by the Europeana Sounds data partners. When a user selects a match between their performance or recording and the contributed archival objects, this connection is recorded in the form of a Web Annotation-compliant link, which can then be stored and later referenced.

Making the Historypin repository into a query-enabled interface by means of a Search API is a crucial step in enabling both the current round-trips of data being designed and piloted on the Europeana Sounds project, but also other future uses of Historypin as a useful and interoperable platform within the Europeana repository generally.

[41] Description of Work 2015-10-19 Europeana Sounds, page 10

These low-level extensions to the Historypin Search API are being established in order to lay the groundwork for the public-facing deployment of these features. The initial Tunepal work has already been deployed, and the use of this technology within Historypin.org is scheduled for 2016.



Pin Details

Returns the pin content.

URL

http://www.historypin.org/en/api/pin/get.json{?parameters}

Parameters

- id (Integer value) - A value representing the User Identification Number (ID)

Example JSON

http://www.historypin.org/en/api/pin/get.json?id=160180

```
{
    "id"                : 160180,
    "caption"           : "Bulgarian Army Theater",
    "description"       : "This is a photo of the famous Bulgarian Army Theater .",
    "pinner_type"       : "photo",
    "type"              : "photo",
    "type_id"           : 0,
    "times_viewed"      : 118,
    "times_favourited"  : 0,
    "order_index"       : 0,

    "display"           : {

            "content"   : "/services/thumb/phid/160180/dim/1000x1000/quality/90/",
            "class"     : "image-pin"

    },

    "oembeded"          : false,
    "date"              : "2014-02-02",
    "date_added"        : "Mon 03 Jun 2013",
    "date_taken"        : "2 February 2014",
    "date_taken_range"  : "2014:2014",

    "location" : {
        "lat"               : 42.694653,
        "lng"               : 23.328998,
        "zoom"              : 15,
        "range"             : 0,
        "raw"               : "",
        "geo_tags"          : "ulitsa &quot;Georgi. S. Rakovski&quot; 98-100, 1000 Sofia, Bulgaria",
        "for_streetview"    : true,
        "streetview" :
        {

            "heading"   : -448.966,
```

Figure 17: A page of Historypin's API documentation showing the Pin Details

Historypin's API documentation can be found at:
http://www.historypin.org/resources/docs/api/site/index.html.

4.4 Roadmap

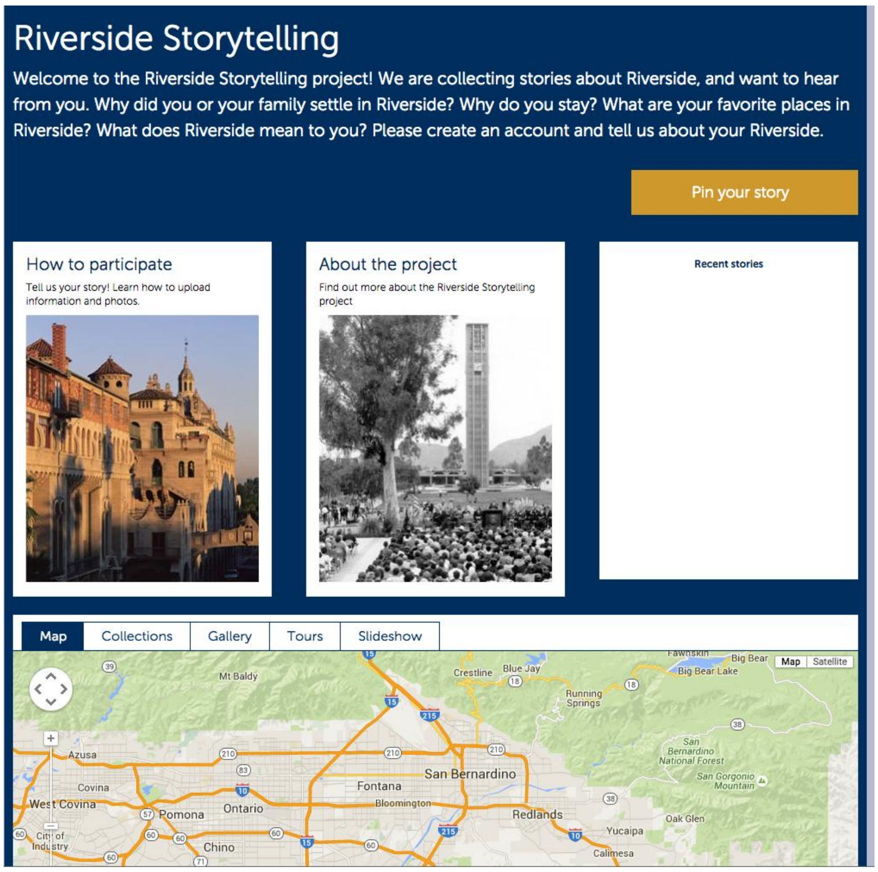
The Tunepal widget has been created and will be updated during the coming months. In parallel, Historypin for Trad tools are being developed. The overall roadmap for the Historypin traditional music pilot is as follows:

Table 2: Roadmap for the Historypin traditional music pilot

Timeframe	Description of work	Status
May - June 2015	Developing Tunepal to allow the widget to connect to it	Complete
July - August 2015	Connecting the Tunepal widget to the Europeana API	Complete
10-16 August 2015	Testing the first iteration of the widget in a stand-alone kiosk set up at the Sligo Fleadh	Complete
September 2015	Evaluation of test at the Fleadh	Complete
September - December 2015	Creating Historypin for Trad crowdsourcing tools	In progress
January - April 2016	Deploying Tunepal widget on thesession.org (social platform for tradional musicians) for testigand scoping extensions to the widget. Developing Historypin for Trad further and running engagement events with traditional musicians to generate contributions	Pending
May - July 2016	Evaluation of crowdsourcing tools and contributions	Pending

The traditional music pilot is mostly on track. Because of necessary work on the accuracy of the Tunepal app that was uncovered during the user testing in the summer of 2015 (please see above for more information about the user testing), the integration of the Tunepal widget onto thesession.org has been slightly delayed. For more information, please see ‘Next steps’ below. However, in the meantime the Historypin for Trad crowdsourcing tools have been developed. This includes preliminary work on being able to round-trip annotations, as based on Historypin’s previous work within the Europeana v3 project [42]. It also includes updated user interfaces on Historypin.org (see below for examples).

[42] http://pro.europeana.eu/files/Europeana_Professional/Projects/Project_list/Europeana_Version3/Deliverables/Ev3%20D3.2%20ProductDevelopmentReport.pdf



Riverside Storytelling

Welcome to the Riverside Storytelling project! We are collecting stories about Riverside, and want to hear from you. Why did you or your family settle in Riverside? Why do you stay? What are your favorite places in Riverside? What does Riverside mean to you? Please create an account and tell us about your Riverside.

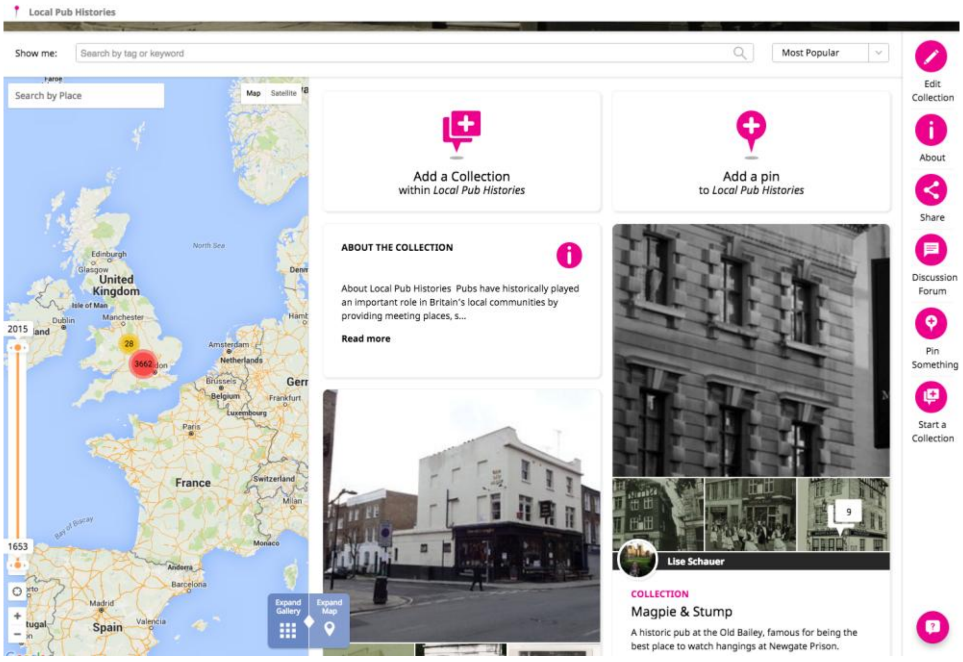
Pin your story

How to participate
Tell us your story! Learn how to upload information and photos.

About the project
Find out more about the Riverside Storytelling project

Recent stories

Figure 18: Example of previous Historypin interface



Local Pub Histories

Show me: Search by tag or keyword · Most Popular

Add a Collection
within *Local Pub Histories*

Add a pin
to *Local Pub Histories*

ABOUT THE COLLECTION

About Local Pub Histories Pubs have historically played an important role in Britain's local communities by providing meeting places, s...

Read more

Lise Schauer

COLLECTION
Magpie & Stump
A historic pub at the Old Bailey, famous for being the best place to watch hangings at Newgate Prison.

Figure 19: Example of updated Historypin interface

The next steps are the integration of Tunepal on thesession.org in spring 2016, as well as testing this integration. The main development here would be a Tunepal search functionality embedded in thesession.org, a feature which does not exist at the moment. Also a more accurate transcription algorithm deployed. In parallel, Historypin's tools for traditional musicians will be further developed and presented to communities already active on Historypin, linking in particular with the new Local Pub History collection.

The roundtripping of annotations to the Europeana API is dependent on the Annotations API being deployed in January 2016, as mentioned in the section above, as well as on the exact way the Tunepal widget will be integrated on thesession.org by thesession.org's developer, as the geolocation data will be generated by thesession.org

4.5 Next steps

Based on the user research, which has confirmed that traditional musicians will use Tunepal as a way of identifying tunes as well as discovering archival material, Historypin can list some next steps, which connect to the roadmap outlined above.

1. Improving Tunepal based on user feedback, especially the accuracy of the music recognition and the usability of the interface. We will update Tunepal based on the feedback received at the Fleadh and will have better transcriptions for banjo, pipes, concertina input by end of December 2015. Whereas the current transcription algorithm relies on continuous monophonic tones which are analysed using a Fast Fourier Transform and then processed using frequency-bounded wavelets, the new algorithm (based on the always advancing state of the art) will have much more sophisticated transient detection to evaluate the harmonics produced by the onset of each note, creating more accurate results for transient-heavy instruments such as banjo and those with richer harmonic series interference such as uilleann pipes and concertina.

2. Creating more links between Tunepal and Europeana. Historypin and Tunepal will discuss possibilities for connecting Tunepal and Europeana based on geo-location of tune queries. The location, date and time of a user querying a tune and selecting a match could be treated as annotations and incorporated in the annotation round-trip work that Historypin has been doing with Europeana as part of Europeana v3. This will not only allow Tunepal to pull in results from the Europeana API, but also to push annotations back, which will improve the metadata of the tunes that have been provided to Europeana and showcase where these traditional archival tunes are still being played around the world.

3. Historypin will deploy the next iteration of Tunepal on www.thesession.org,, a forum for traditional musicians. It can be integrated as a tune recognition tool there. TheSession.org users will be asked to complete a short online questionnaire, after they use the tool, to allow us to gather more data about accuracy, availability of archival records and how the tool is perceived by our target audience.

4. Historypin will do more testing of the Tunepal app, this time with adult musicians, focusing mostly on the usability of the interface and to try and get clearer answers to what value the Europeana search functionality adds to the app, as well as how the app adds value to Europeana Sounds.

5 WITH Platform

The WITH is a culture-sharing crowdsourcing platform that allows Europeana Sounds partners, external cultural institutions and third party developers to easily search for cultural resources with the aim to collect, enrich, share and co-create with other users. Everything users can do in WITH is saved as annotations, using Europeana's WA. WITH platform development is partially supported through Europeana Sounds, Europeana Space [43], EUscreenXL [44] and Europeana Food and Drink [45]. The development of WITH has been divided in four parts. In Europeana Sounds we develop the crowdsourcing functionalities of WITH, which are the creation of collections, exhibitions, and the creation of annotations. One of the main contributions of WITH in the crowdsourcing infrastructure is the user collections API that is planned to be integrated in the Europeana API in July 2016.

WITH is designed and developed in alignment with complementary services from the Europeana core infrastructure, and informed by the design of respective infrastructures being developed such as Europeana Labs [46], the Europeana Cloud [47] and LoCloud [48].

When reaching out to potential user groups and stakeholders, the methods of engagement can be broken down to:

- Discover (leveraging APIs to cultural content from around the world).
- Create (collections, exhibitions and stories).
- Annotate (link and enrich using SKOS thesauri and linked data repositories).
- Participate (share, tag and rate collections, follow users & spaces, join user groups).
- Build (use the API to access data and services).

The platform consists of the following components:

- The Storage layer for metadata and content.
- The Aggregation Workflow Engine for managing and publishing DCH resources.
- The Processing Infrastructure and the Services deployment and integration environment.
- The User Interface to access functionalities and visualize data.
- The Access APIs for the platform's data and services.

[43] http://www.europeana-space.eu/
[44] http://www.euscreen.eu/
[45] http://foodanddrinkeurope.eu/
[46] http://labs.europeana.eu
[47] http://pro.europeana.eu/project/europeana-cloud
[48] http://www.locloud.eu/

Furthermore, the analysis is informed by evolutions in the Europeana ecosystem, especially in relevant projects such as Europeana Space, Europeana Creative [49], LoCloud, Europeana Food & Drink and EUScreenXL.

5.1 Validation of the requirements

Participation in Europeana's Projects Group Assembly and related technical workshops as well as in the EuropeanaTech R&D community [50] and its respective task forces has also assisted in gathering requirements from researchers, developers and experts in the network. The result was a more detailed specification of the WITH's architecture and the identification of potential synergies and cooperation with existing and under development systems, in various technical levels such as storage infrastructure or third party services for developers and users. Interoperability with the Europeana ecosystem is a significant aspect of the architecture in order to inform technical choices with existing system capabilities and requirements.

5.2 Backend

Items coming from different sources are stored as records in the data infrastructure. A record comprises the content of the item and the metadata which describes it in a well-defined JSON format. The WITH JSON includes a number of important fields, such as "title", "description", "dataProvider", "rights" etc, which are extracted from the original item metadata. Custom mappers from a number of supported schemes to the WITH-JSON can be easily added to the system, while for datasets imported from MINT, the mapping editor can be used. In all cases, the serialization(s) of the full metadata representation(s) of the original item are retrieved and saved in the WITH JSON under the "content" field. A number of different schemes, either in XML or JSON format, are supported, like Europeana's JSON-EDM [51] and JSON-LD [52], EDM XML [53] or LIDO XML [54].

A record saved in the WITH database can only exist as part of a Collection, which is the first-class data structure in WITH, and corresponds to a set of records along with a set of metadata describing the collection itself (currently these include the fields title and description, a set that will evolve in the future). Users of the platform can search for records they are interested in (both from the external data pools and the WITH database), and make a selection to create their own collections. Users can manage their collections by adding records, and editing their title and description. They can also share their collections with other users or user groups, by passing them READ or WRITE access.

An Exhibition is a type of collection that contains some extra annotations (descriptions, attached video) and a special way of visualizing it on the front-end. Via the exhibition editor, users can select records from their collections and enrich them with additional description and videos to "tell a story". Other

[49] http://pro.europeana.eu/structure/europeana-creative
[50] http://pro.europeana.eu/get-involved/europeana-tech
[51] http://labs.europeana.eu/api/record
[52] http://labs.europeana.eu/api/record-jsonld
[53] http://pro.europeana.eu/page/edm-documentation
[54] www.lido-schema.org

types of collections and ways to visualize them will be supported in the future to serve the needs of different types of users, e.g. preparing a presentation for educational purposes.

Records, collections and exhibitions are saved as Mongo documents in the document-based MongoDB [55] maintained by the WITH platform. The Jackson JSON processor is used to convert the JSON representations into Java entities, and the Morphia library to map them in turn into the Mongo DB and back in a typesafe way. Morphia's Query API [56] is used to search in the database, which supports filter criteria, offset specification, and limiting of the number of results.

The platform maintains a distinction between the concept of an “external item”, i.e. the original object and its metadata as it is in the external source it comes from (e.g., Europeana items, a MINT dataset, or user contributed content), and the copies of the item saved in the database as part of a specific collection. Each time an item is “collected” by a user (i.e. added to a collection), a new object of type Record is saved in the WITH database. If the item selected by the user does not already exist in the WITH database (i.e. is the result of a search on an external resource), the full metadata of the item – if available – are retrieved from the external source, and parsed to extract the WITH JSON fields. When a record referring to the same external item already exists in the local database, then a new Record instance is created by copying the WITH JSON fields that refer to the common “external item” metadata. The record instance contains fields that associate it with the collection it belongs to and its position in it. This way, the WITH JSON contains “versions” or “copies” of the same “external item”, each of which follows its own “lifecycle” from the point it becomes part of a collection: users may put extra annotations on the record object (e.g. tags, extra descriptions, attach videos to it etc), edit annotations added by other users if they have the necessary access level (e.g. collaborative management of a collection), rearrange its position in an Exhibition etc. All this while provenance to the original record is maintained.

Published datasets can also be available through a semantic store, an industrial-strength repository following W3C recommendations for RDF serializations and the SPARQL 1.1 Query Language . Having reviewed and tested several solutions for RDF storage such as graph databases and triple-stores, among them Bigdata, OWLIM , 4Store , Neo4j , SHARD , Dydra and Sesame , WITH uses the Apache Jena TDB store and Fuseki SPARQL server . The setup provides a high performance RDF store and a server that offers REST-style SPARQL HTTP Update, SPARQL Query, and SPARQL Update. It uses the SOH (SPARQL Over HTTP) set of scripts for working with SPARQL 1.1. The transformation between XML and semantic web serializations for datasets that are represented using RDF vocabularies are performed by the system, transparently to the user.

5.3 Application Programming Interfaces

The WITH back-end uses a REST API to communicate with the front-end and provide a way for developers of other applications to use the data and services that WITH provides. WITH is a complex tool oriented towards both users and developers. Therefore all calls, schemas and responses have been documented in detail and are provided in the WITH developers page. This documentation is encoded in the Swagger schema (detailed in http://swagger.io/specification/) and provided in two visualizations,

[55] https://www.mongodb.org/
[56] https://mongodb.github.io/morphia/

the default Swagger UI and the Swagger UI responsive theme (https://github.com/jensoleg/swagger-ui). These pages can be accessed through links in the landing page of WITH.

5.4 Frontend: crowdsourcing functionalities

5.4.1 Collections and exhibitions

This section presents the crowdsourcing functionalities of WITH, which are the creation and management of collections and exhibitions, the sharing and co-creation of collections by many users and the creation of user-group spaces.

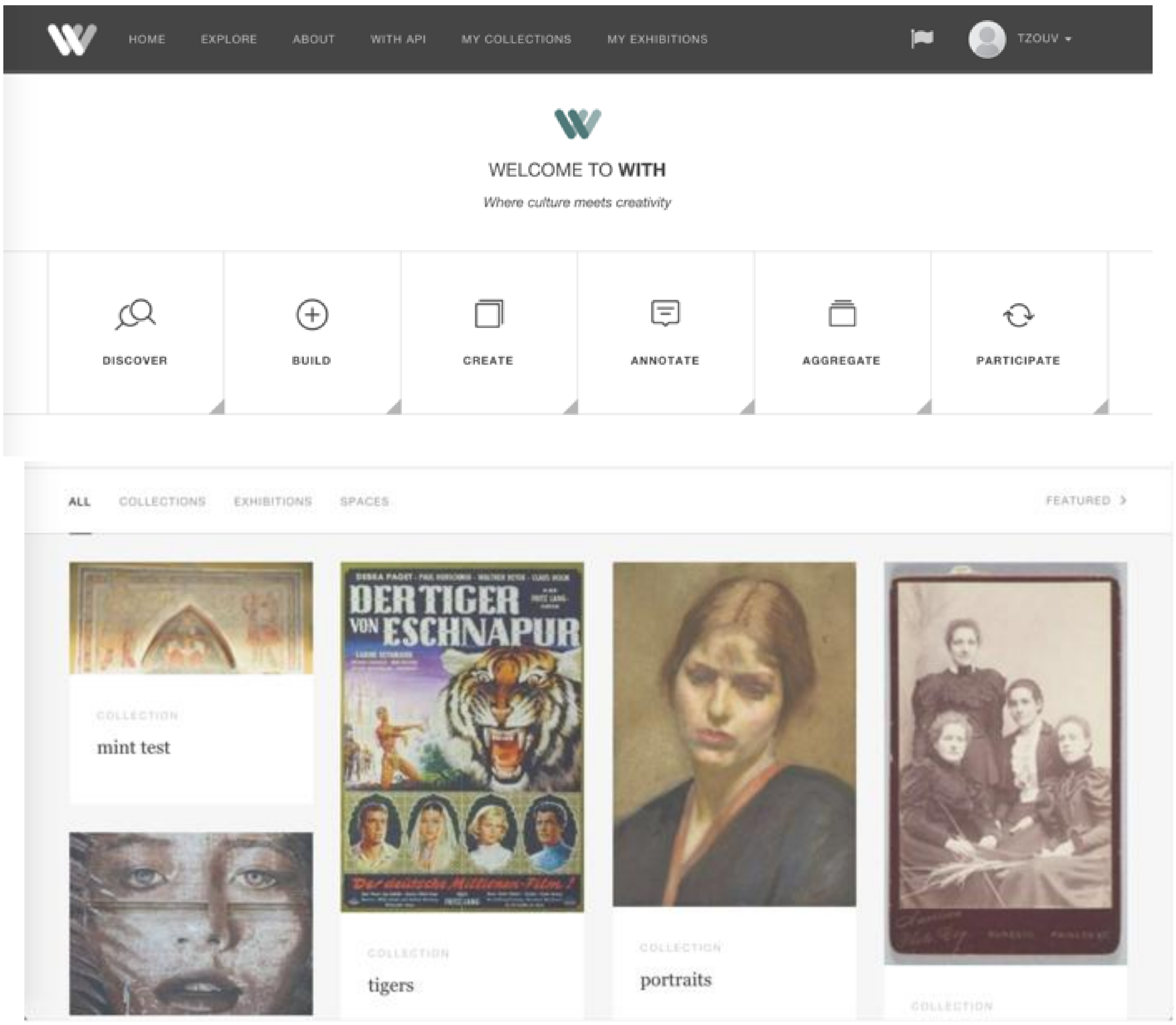


Figure 20: Landing page

5.5 Identification, authentication, rights management

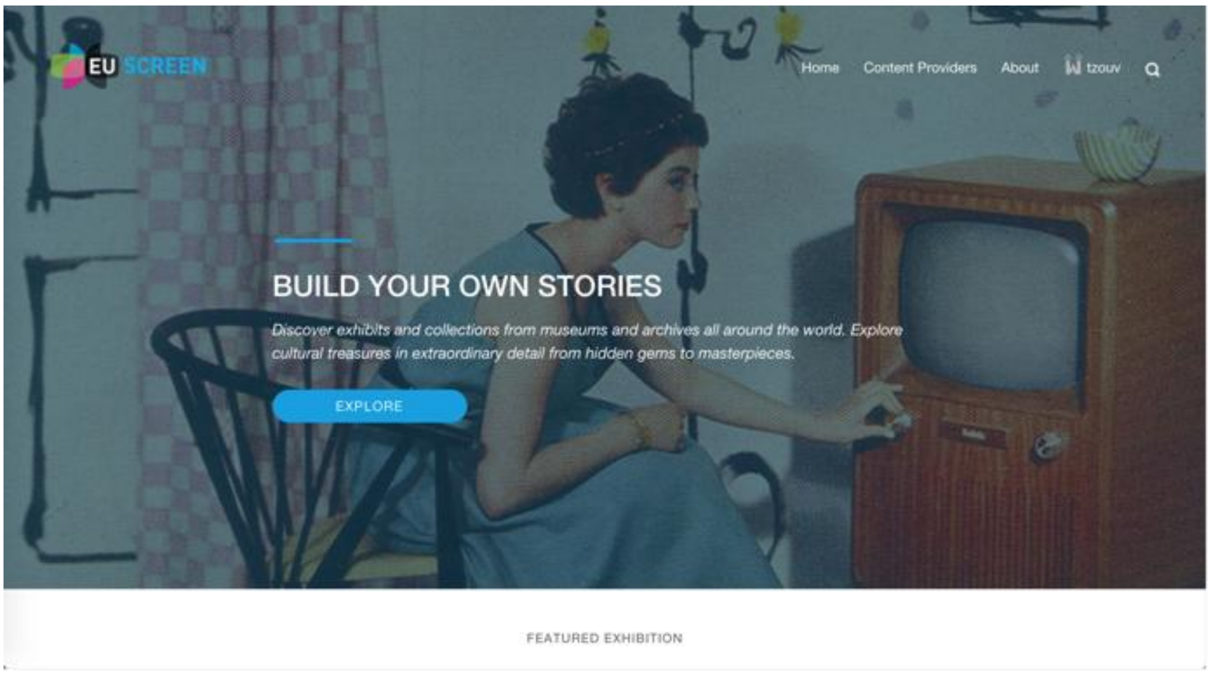
The platform employs a typical user management system to authenticate access by identifying a user according to details of his account. It also implements user groups e.g. an organization or an interest group, for which access rights are configured and assigned collectively. WITH uses access rights to control access to specific areas of the repository. This is accomplished by assigning privileges to either allow or deny access to a resource (dataset, content or service).

5.5.1 Spaces

Every Space is practically a user group that contains a list of users along with some metadata for the group itself. Moreover, it contains its creator, a list of administrators and a list of parent groups. The creator has the role of “superuser” for the group and the rights to edit the metadata of the group, add/remove administrators or users of the group and even delete the group. The administrators of the group is a list of users. Their main job is to add and remove users of the group.

There are basically three types of user groups:

- Interest groups (e.g. Mozart lovers): general purpose groups that do not correspond to a specific organization or project. For example a group of users who admire a specific art form (e.g. Classical Music).
- Organizations (e.g. NTUA, NISV): Organizations are an extension to the basic User Groups containing more metadata (e.g. address) for their description. They may be museums, universities, research institutes etc.
- Projects (e.g. Europeana Sounds): Projects are an extension to the basic User Groups (as Organizations) which usually have the organizations as children.



BUILD YOUR OWN STORIES

Discover exhibits and collections from museums and archives all around the world. Explore cultural treasures in extraordinary detail from hidden gems to masterpieces.

FEATURED EXHIBITION

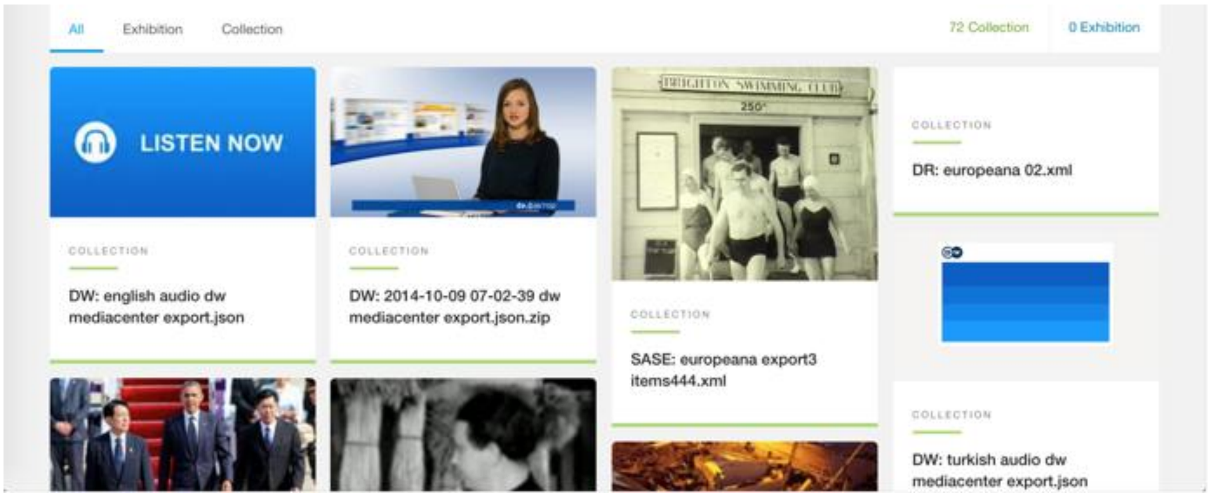
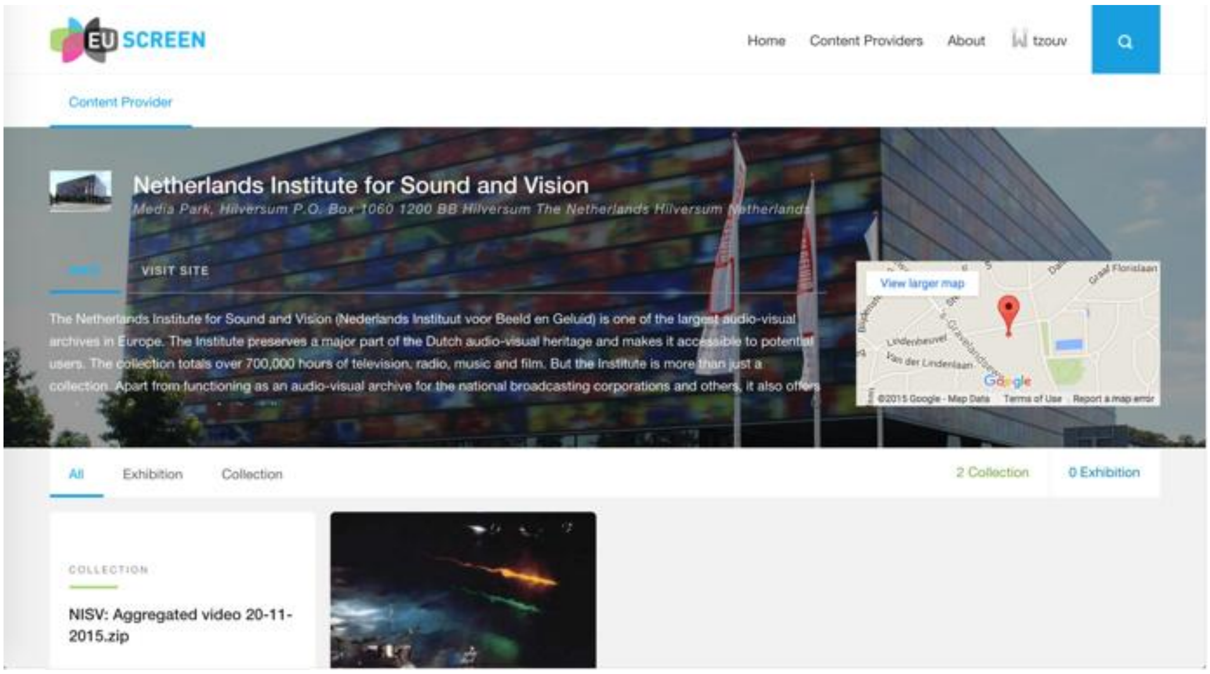

All
Exhibition
Collection
72 Collection
0 Exhibition
LISTEN NOW
COLLECTION
DW: english audio dw mediacenter export.json
COLLECTION
DW: 2014-10-09 07-02-39 dw mediacenter export.json.zip
COLLECTION
SASE: europeana export3 items444.xml
COLLECTION
DR: europeana 02.xml
COLLECTION
DW: turkish audio dw mediacenter export.json

Figure 21: The EUscreenXL project space



Netherlands Institute for Sound and Vision
Media Park, Hilversum P.O. Box 1060 1200 BB Hilversum The Netherlands Hilversum Netherlands

The Netherlands Institute for Sound and Vision (Nederlands Instituut voor Beeld en Geluid) is one of the largest audio-visual archives in Europe. The Institute preserves a major part of the Dutch audio-visual heritage and makes it accessible to potential users. The collection totals over 700,000 hours of television, radio, music and film. But the Institute is more than just a collection. Apart from functioning as an audio-visual archive for the national broadcasting corporations and others, it also offers

Figure 22: User/Organization provider page

5.6 Workflows and visualizations

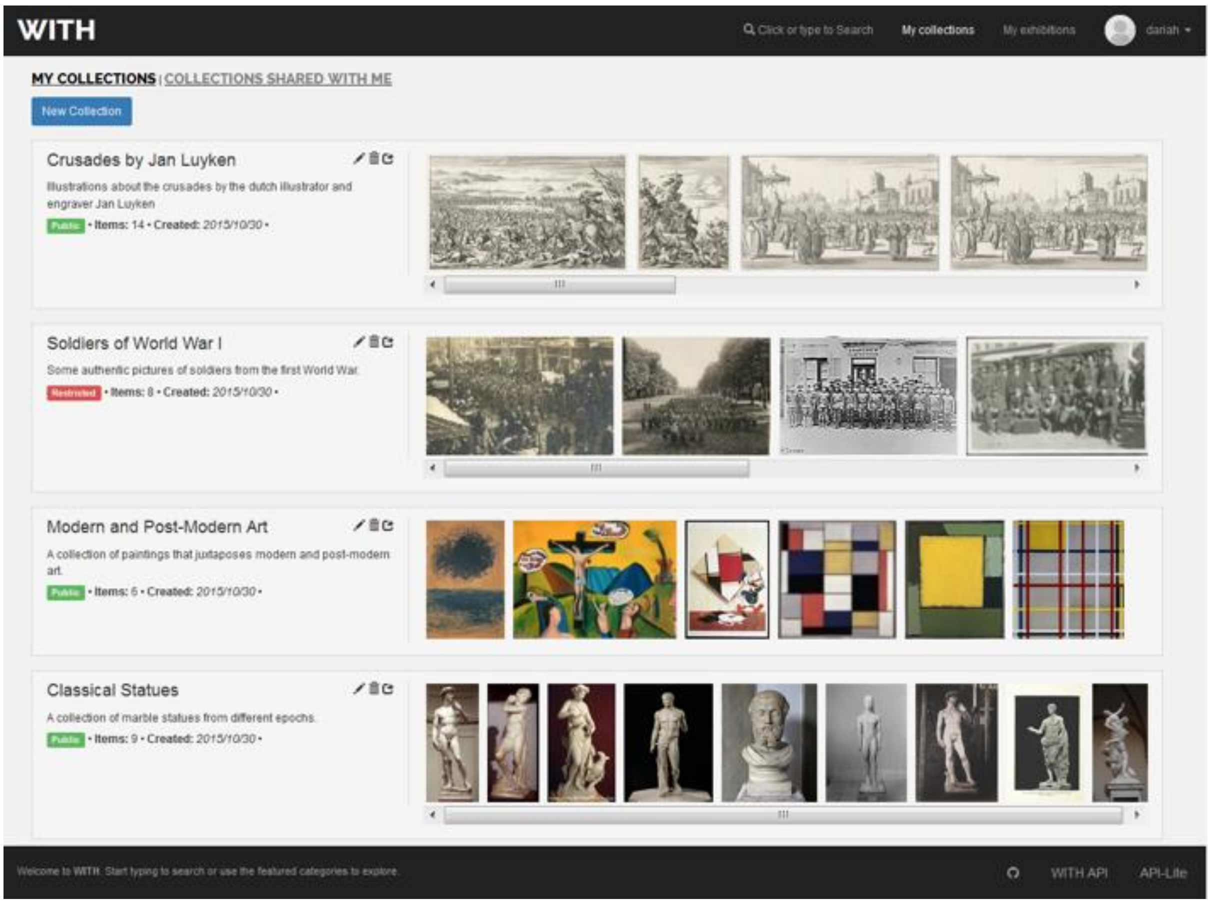
WITH is addressing the needs of a wide spectrum of users that engage with it in different crowdsourcing scenarios. It can be used by the public, domain scholars or content providers, being members of a network, a project, or participating in events like edit-a-thons or hackathons. Depending on the type of user and the intended usage there are several workflows that can be followed for populating with content, creating and managing collections, and consuming the produced work. WITH introduces a collection management front end to leverage the capabilities that its Data Infrastructure and APIs offer for content sourcing, editing and curation and, to set the space for collaborative creation and re-use of cultural heritage content.

The user interface has been implemented as a Single Page Application (SPA), with the aim to offer a fluid and responsive app-like experience to the user, without the need for constant page reloads. The SPA experience is powered on top of the Play web framework for communicating with the REST API back-end. The Play framework provides a lightweight and scalable architecture for building web applications following the model–view–controller (MVC) architectural pattern.

The model object layer consists of a set of Java classes that represent the data structures and operations on which the application operates. The controller is the intermediate layer between the model and the user interactions with the browser, and is responsible for responding to HTTP requests by invoking actions that access or modify the data model. A clear RESTful model for designing an API allows the connection between HTTP methods and the respective calls to the methods that perform the necessary actions on the backend. The view layer renders the data model into the form of user interface. The knockout JavaScript library[57] is used to build the interfaces that correspond to the underlying data objects, through declarative bindings. This way, whenever the data model's state changes, e.g., a new collection has been created or deleted, the UI updates automatically.

Users can register at the platform or use their Facebook and g+ accounts to log in. They have a personal space for datasets they are managing, in the different forms offered such as collections and exhibitions. The landing page showcases featured collections, exhibitions and spaces together with a list of public datasets. Using the MINT upload, the direct upload functionality or the federated search, a user can create and manage collections, visualize and eventually share them publicly or to a specific user group or space.

[57] http://knockoutjs.com/



Crusades by Jan Luyken

Illustrations about the crusades by the dutch illustrator and engraver Jan Luyken

Public • Items: 14 • Created: 2015/10/30 •

Soldiers of World War I

Some authentic pictures of soldiers from the first World War.

Restricted • Items: 8 • Created: 2015/10/30 •

Modern and Post-Modern Art

A collection of paintings that juxtaposes modern and post-modern art.

Public • Items: 6 • Created: 2015/10/30 •

Classical Statues

A collection of marble statues from different epochs.

Public • Items: 9 • Created: 2015/10/30 •

Welcome to WITH. Start typing to search or use the featured categories to explore.

Figure 23: My Collections screen

Users can upload their own content and provide metadata for it using the functionality of the front-end or, for batch upload in the case of content providers, use the Mint to ingest metadata records, align them to the repository using the mapping editor, transform and publish them as collections. The Data Infrastructure is the centrepiece of the platform, being in charge of storing all required versions of a dataset and exposing them through the API according to license and terms of use. The interaction with the stored content happens through the UI for normal users and the APIs for developers. A user can discover content from external sources as was illustrated in the previous section and manage it using the front-end. The results of the federated search can be browsed individually and collected.

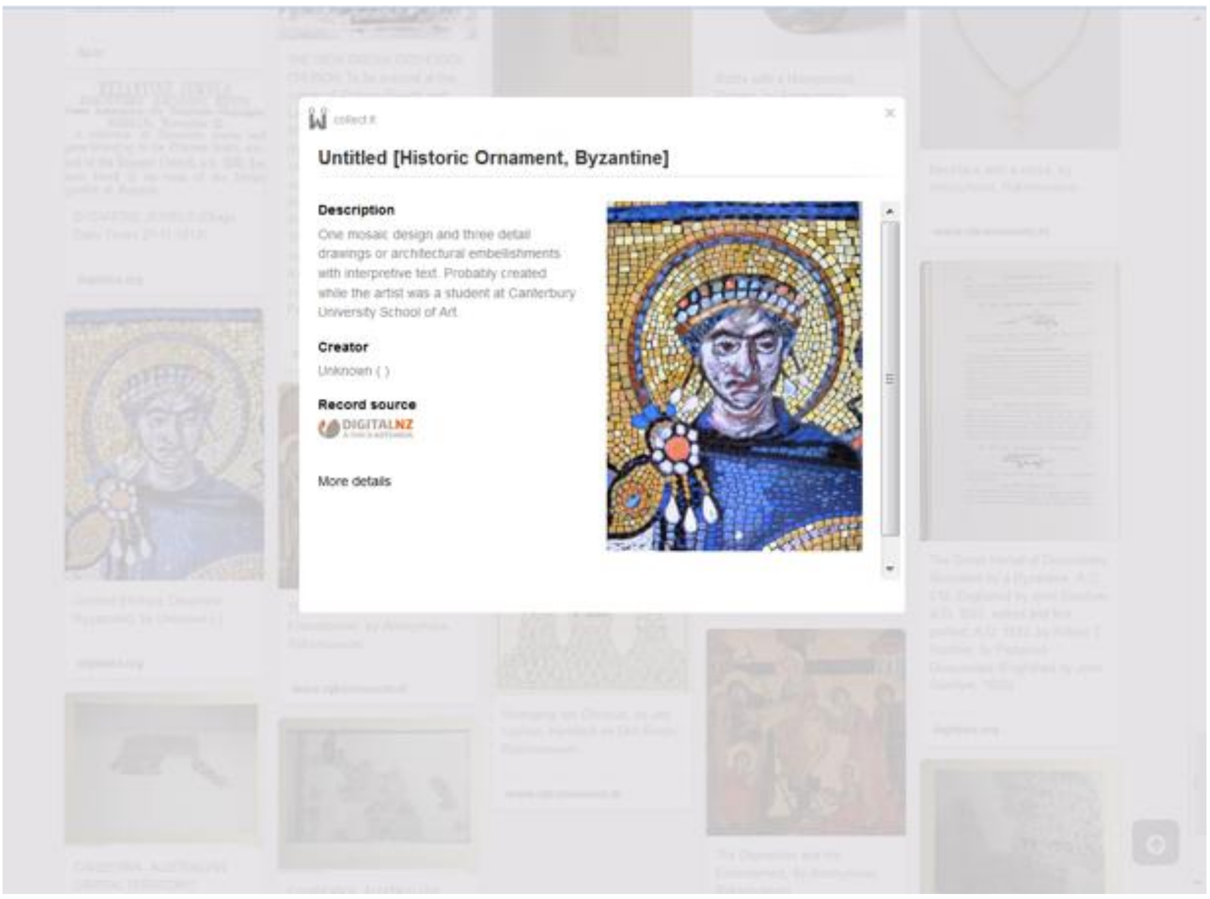

collect it
Untitled [Historic Ornament, Byzantine]
Description
One mosaic design and three detail drawings or architectural embellishments with interpretive text. Probably created while the artist was a student at Canterbury University School of Art
Creator
Unknown ( )
Record source
DIGITALNZ
More details

Figure 24: Viewing details for a resource from an external repository

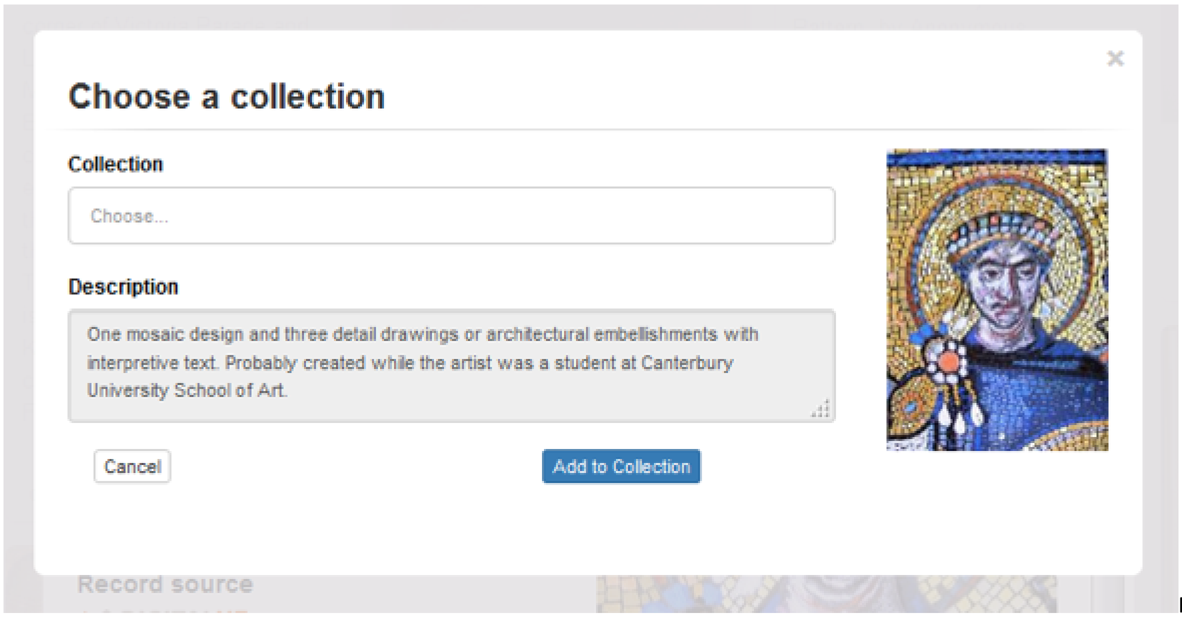
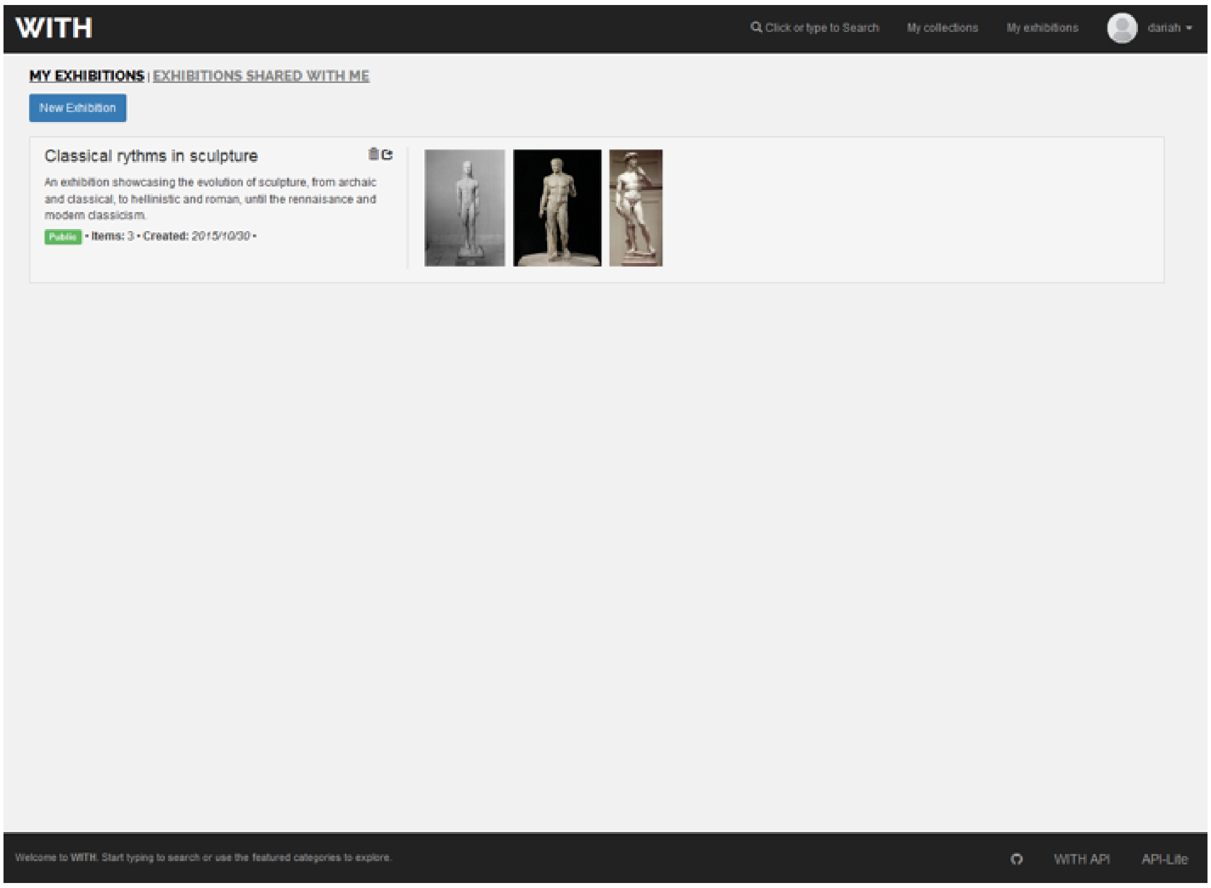

Choose a collection
Collection
Choose...
Description
One mosaic design and three detail drawings or architectural embellishments with interpretive text. Probably created while the artist was a student at Canterbury University School of Art.
Cancel
Add to Collection
Record source

Figure
25: Collecting a resource from an external repository

Other visualization and organization options for datasets are being explored that help contextualize a dataset and visualize it with the purpose of sharing a personal experience around the content that consists it e.g. telling a story. The exhibitions UI is a step towards this direction, allowing users to add information (captions, descriptions, embedded multimedia) for each resource in a collection and use its play out method to present it in a more engaging manner. Similarly we are investigating organization and visualization schemes such as 3D galleries, user or content provider pages and spaces.



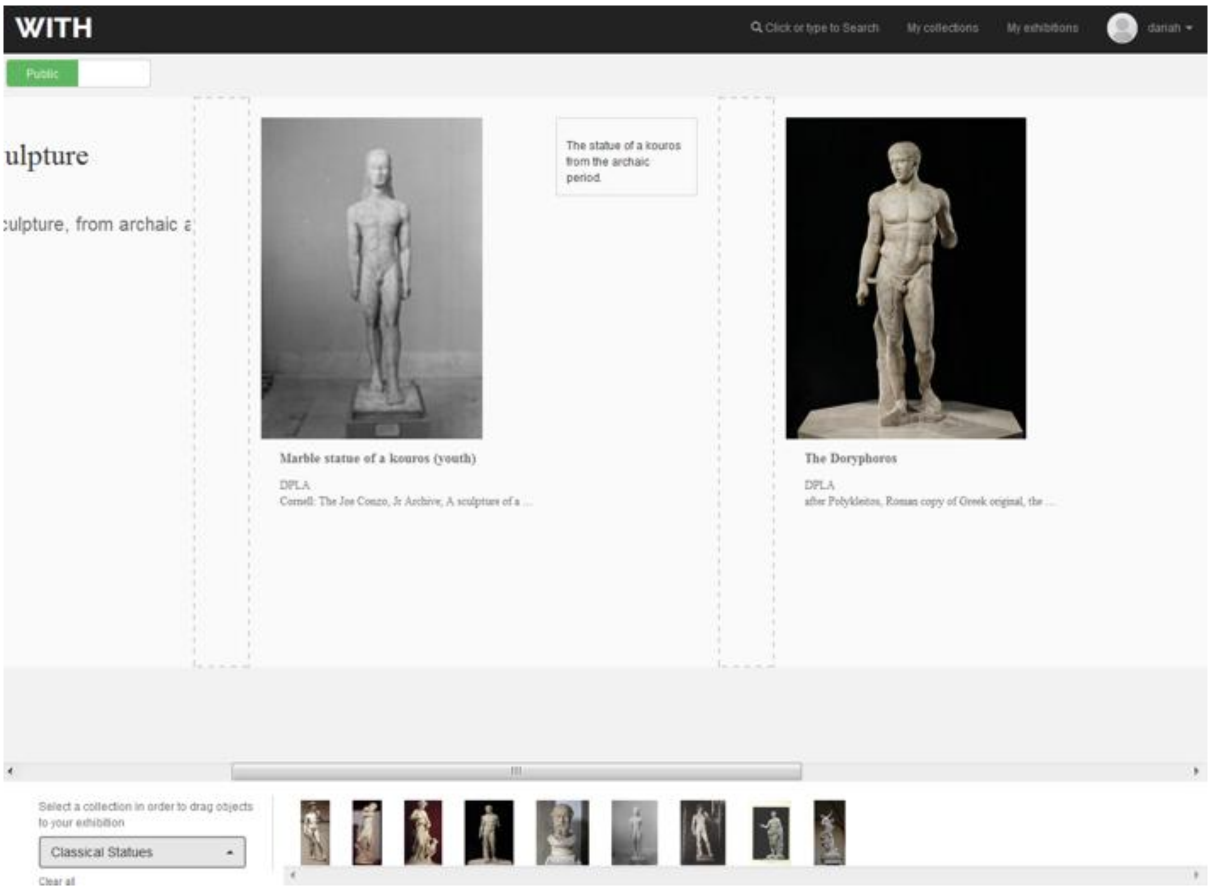
Classical rythms in sculpture

An exhibition showcasing the evolution of sculpture, from archaic and classical, to hellinistic and roman, until the rennaisance and modern classicism.

Public • Items: 3 • Created: 2015/10/30 •

Welcome to WITH. Start typing to search or use the featured categories to explore.

Figure 26: My Exhibitions screen



ulpture

:ulpture, from archaic a

The statue of a kouros from the archaic period.

Marble statue of a kouros (youth)
DPLA
Cornell: The Joe Conzo, Jr Archive, A sculpture of a ...

The Doryphoros
DPLA
after Polykleitos, Roman copy of Greek original, the ...

Select a collection in order to drag objects to your exhibition
Classical Statues
Clear all

Figure 27: Exhibition editor

The latter is a concept that was introduced with user groups and, in practice, corresponds to specific, access-based views of stored data. A user group has access to collections shared by its members and can be moderated. WITH offers the ability to visualize user groups individually and allows customization of the front end (descriptive texts, images, CSS). The content (collections, exhibitions etc.) is limited to those that the user group can access and the scope of the search engine can be customized as well, for example to exclude some sources or, to only search for video resources etc.

5.7 Social dimension

There are several features that introduce and reinforce the social dimension of the platform, focusing on collaborative creation and offering the ability to easily share and disseminate collections and their updates. A user can make a collection public for other users of the platform or join and share it with a user group, as those were defined in the previous chapter. A basic rating mechanism in the form of favourites/likes for platform resources (items, collections, user groups etc.) is also being tested to provide the basis for a ranking engine that will improve search results and allow the search engine to adapt to user preferences. Finally, a user can join or follow user group or project spaces and other users, with a notification mechanism that informs her of updates, pending requests or invitations.

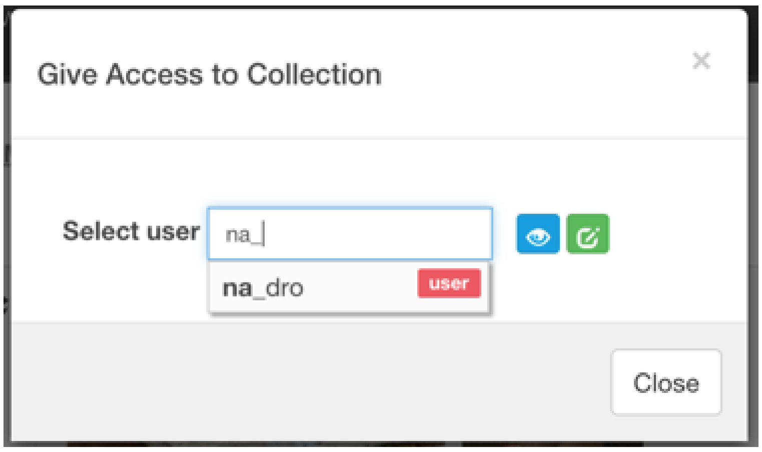
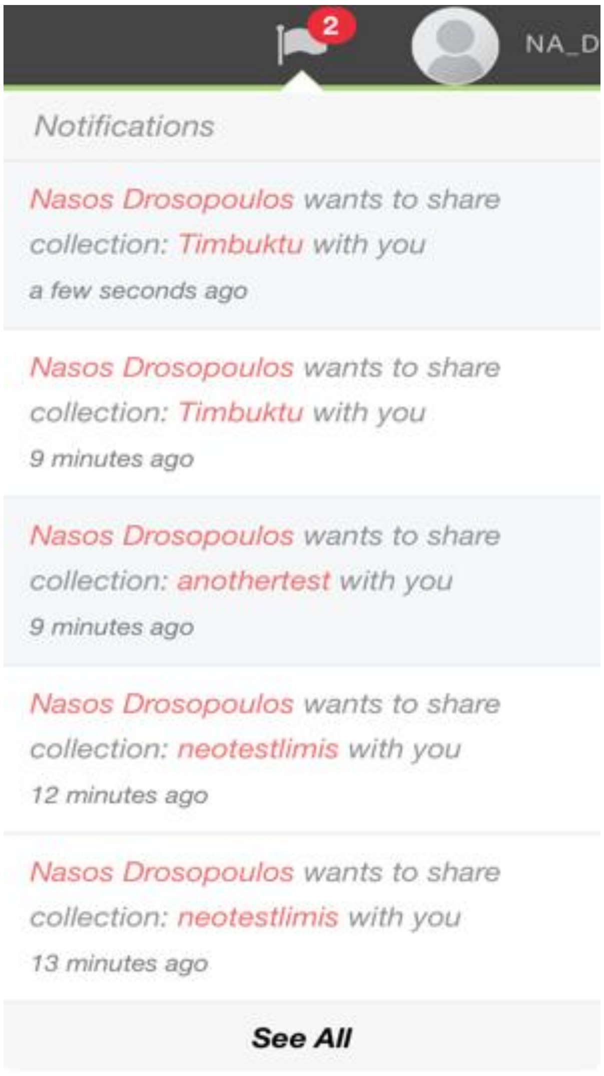


Figure 28: Sharing a collection with a user or group



Nasos Drosopoulos wants to share collection: Timbuktu with you
a few seconds ago

Nasos Drosopoulos wants to share collection: Timbuktu with you
9 minutes ago

Nasos Drosopoulos wants to share collection: anothertest with you
9 minutes ago

Nasos Drosopoulos wants to share collection: neotestlimis with you
12 minutes ago

Nasos Drosopoulos wants to share collection: neotestlimis with you
13 minutes ago

Figure 29: Notifications

5.8 Next steps

The current set of crowdsourcing functionalities of the WITH platform include the creation of collections, exhibitions and spaces. The sharing and co-creation functionalities are also enabled to allow users co-create collections and exhibitions. The aim of the crowdsourcing functionalities are to create and save annotation in the Europeana's annotation API. The next plans include the development and finalisation of the enrichment module that will allow users to enrich Europeana items using external thesauri, authority files and linked open data sources. It is expected to have the first version of the enrichment module in March 2016. The following table shows the connection of the WITH functionalities with the defined crowdsourcing user stories[58].

Table 3: WITH functionalities roadmap

User story	Epic	Description	Prio	Deadline
S11	E5	Technical requirements for editing or deleting own annotations	MH	March 2016
S17	E3	Create collections of related items to be used in third party applications	MH	January 2016
S21	E3	Add simple tags connecting objects related to a given thematic. Simple tags must have a type	MH	March 2016
S33	E8	Create user collections by providing a set of metadata regarding the purpose of the collection	MH	January 2016
S34	E8	Create a user collection that will aggregate only sound content	MH	January 2016
S37, S38, S42	E8	Publishing (private) user collections	MH	January 2016
S41	E8	Define multilingual labels for User Created Collections	MH	March 2016
S43	E8	Changing the visibility of the User Collections between private/group/public	MH	January 2016
S36, S40, S44, S47	E8	Add Europeana Object to User Collections that can be accessed only by a given group of users	NTH	January 2016

6 The enrichment annotation scenario pilot: Pundit

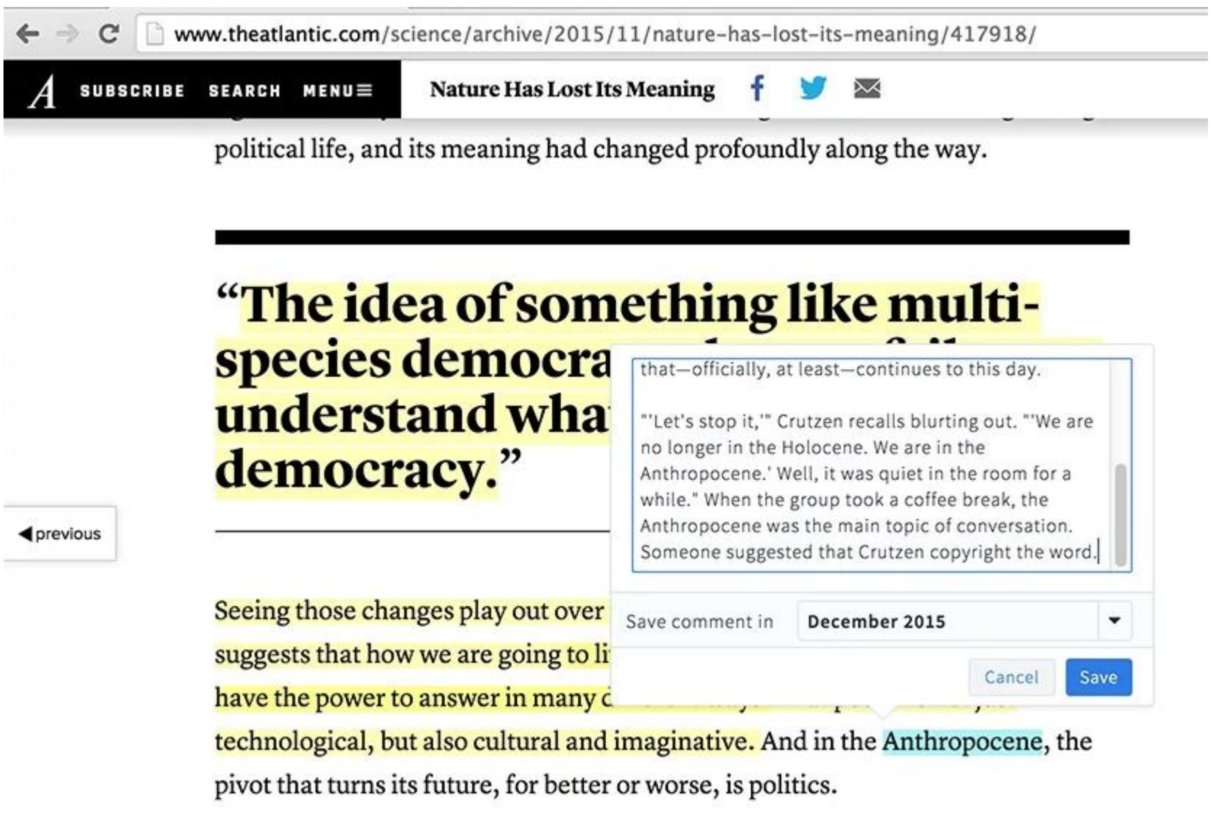
6.1 The new version of the Pundit client

After the analysis of the results obtained from the first user test performed using crowdsourcing platform built with the tool Pundit, we released a new[59] version of the Pundit tool[60] which is the main

[58] Refer to D2.2, MS11 and the following working spreadsheet document: https://docs.google.com/spreadsheets/d/11Rh0aR3eikc_A-k1R0nVp3WW80dL4T_J4IRFpq_7cdU/edit#gid=1667815154
[59] http://thepund.it/new-pundit-annotator-is-out-web-annotation/

instrument of the widgets used in the context of Europeana Sounds. The new version of Pundit (Pundit 2) has a more intuitive interface and is much more user-oriented.

The new version of the client has been released as a Chrome extension but it can also be used as an embedded component exactly, as a widget used for the scenario of manual annotation in the context of the Europeana Sounds project.


www.theatlantic.com/science/archive/2015/11/nature-has-lost-its-meaning/417918/

SUBSCRIBE SEARCH MENU Nature Has Lost Its Meaning

political life, and its meaning had changed profoundly along the way.

"The idea of something like multi-species democra understand wha democracy."

◀previous

that—officially, at least—continues to this day.

"'Let's stop it,'" Crutzen recalls blurting out. "'We are no longer in the Holocene. We are in the Anthropocene.' Well, it was quiet in the room for a while." When the group took a coffee break, the Anthropocene was the main topic of conversation. Someone suggested that Crutzen copyright the word.

Save comment in: December 2015

Cancel Save

Seeing those changes play out over suggests that how we are going to li have the power to answer in many c technological, but also cultural and imaginative. And in the Anthropocene, the pivot that turns its future, for better or worse, is politics.

Figure 30: This image shows a web page annotated (simple comment) with the new version of Pundit.

Two different versions have been released:

- Annotator: This version is a simplified version of the client with limited features (at the moment comment and highlight)
- Annotator Pro: This version contains the features implemented in "Annotator" together with all previous Pundit features (semantic annotation, triple composer, etc.) revised with new graphics

The new version of the widget developed for Europeana Sounds is designed as an embedded version of the client "Annotator Pro".

[60] Both server and client. For the new version of the client refer to http://thepund.it/annotator and http://thepund.it/annotatorpro

Together with the development of new versions of the client Net7 released a new version of the server and a platform for the management of records created through the use of Annotator or Annotator Pro.

Its internal architecture of the new version of the server is organized in multiple layers:

- The API Layer is the front-end for applications. It is based on the Apache Jersey framework[61]
- The permission layer verifies that the request arrives from authorised users, using, as explained below, the OAUth 2.0 protocol[62].
- The Query & Persistence layer: it is the true heart of the Pundit Annotation Server. It manages data retrieval and persistence.

The main features of the platform (Annotation Manager) are:

1. Login using the same credentials ("Single Sign On" based on the OAuth 2.0 protocol) used to access the Pundit client.
2. Browse and search through the records of a user's "notebook". A "notebook" is a collection of annotations created by a user.
3. Create new notebooks, edit or delete.
4. Perform a global search of all annotations created.

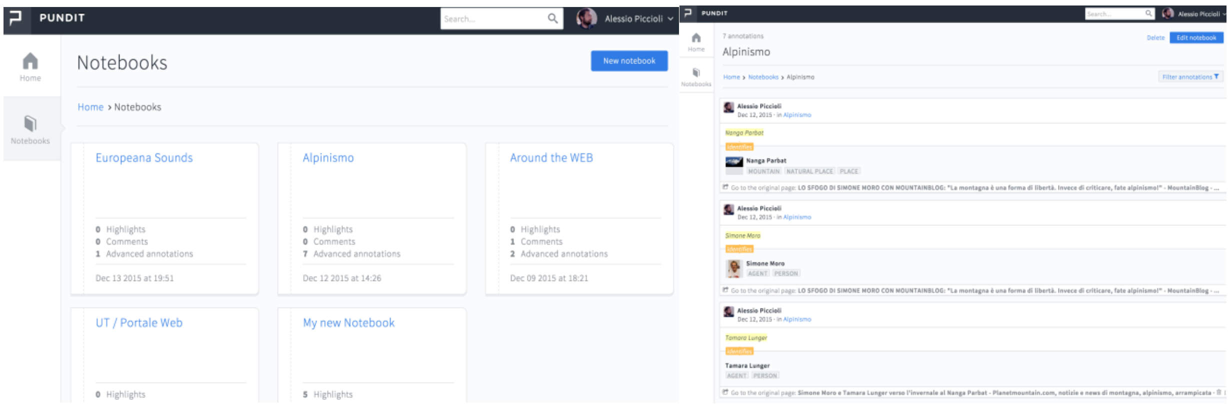


Figure 31: On the left: the list of notebooks. On the right: the annotations of a notebook

The second pilot with MIMO

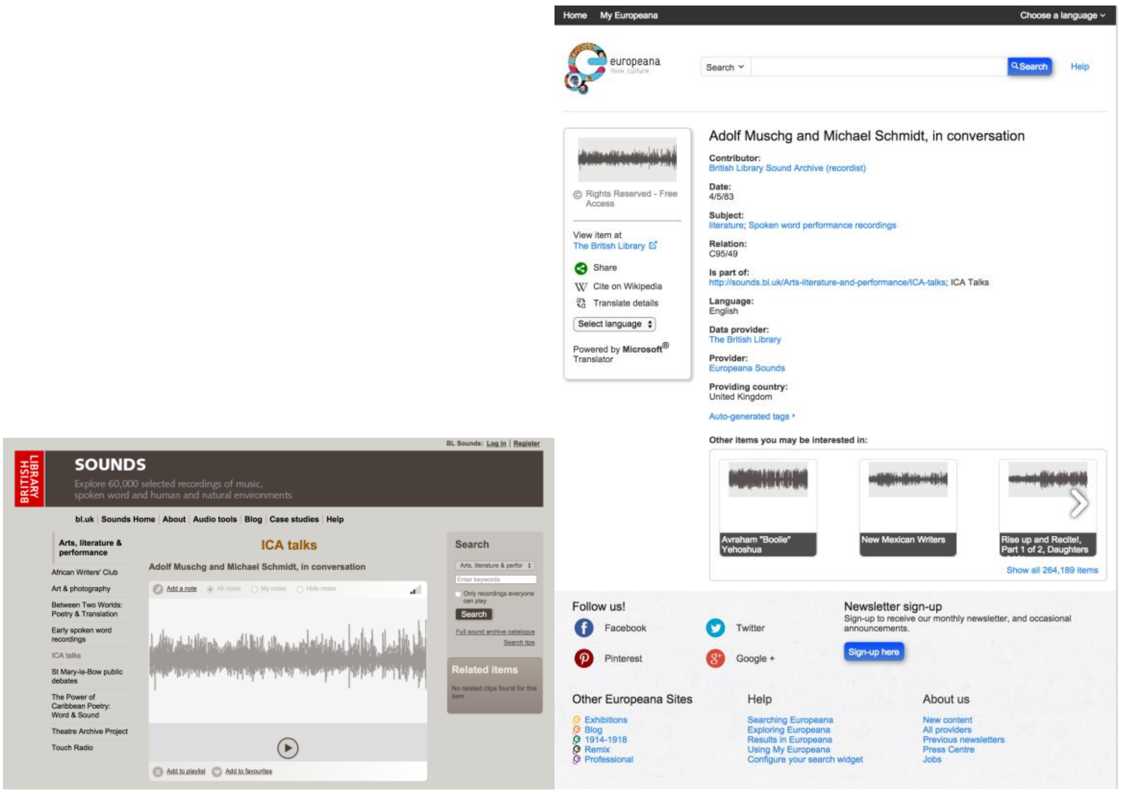
One of the new features of the Pundit client is the ability to create semantic annotations that create links between the resources of Europeana (CHO) and controlled-vocabulary terms from the MIMO Vocabulary and Thesaurus[63]. MIMO is the world's largest freely accessible database for information on musical instruments held in public collections: at this moment it contains 55,535 records of instruments translated in seven different languages (EN, FR, IT, DE, NL, SV, CA) and accessible through Open Data standard protocols (SPARQL). This new Pundit feature has been realised by creating a new connector in the resource panel that directly queries the SPARQL-endpoint[64] of the MIMO project. This feature is the basis of the new pilot experiment within the Europeana Sounds project: It allows user to enrich

[61] https://jersey.java.net/
[62] http://oauth.net/2/
[63] http://www.mimo-international.com/MIMO/
[64] http://data.mimo-db.eu:9091/sparql/data

metadata records that describe music-related items with structured data from the MIMO Vocabulary and Thesaurus. Enrichments with these MIMO terms have two direct benefits: They allow for items to be linked across collections (1), and also create multi-lingual access points for these items (2). The idea for this MIMO pilot originated from the WP2 workshop during the 2nd training session in Athens, where a lot of data providers indicated interest in enriching their records with MIMO terms. The pilot will be launched at the end of January 2016.

6.2 The new version of the widget

The widget created by Net7 allows you to view and enrich the metadata imported from Europeana, starting the user flow directly from the website of the content provider. Based on the CHO's identification code in the Europeana platform, the widget renders a simple view of the resource within an interface that allows the user to start annotating.



Adolf Muschg and Michael Schmidt, in conversation

Contributor:
British Library Sound Archive (recordist)

Date:
4/5/83

Subject:
literature; Spoken word performance recordings

Relation:
C95/49

Is part of:
http://sounds.bl.uk/Arts-literature-and-performance/ICA-talks; ICA Talks

Language:
English

Data provider:
The British Library

Provider:
Europeana Sounds

Providing country:
United Kingdom

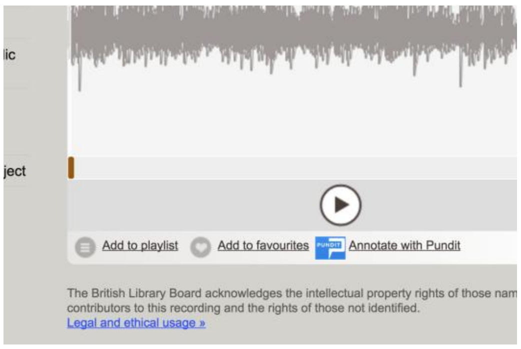
Auto-generated tags

Other items you may be interested in:

ICA talks

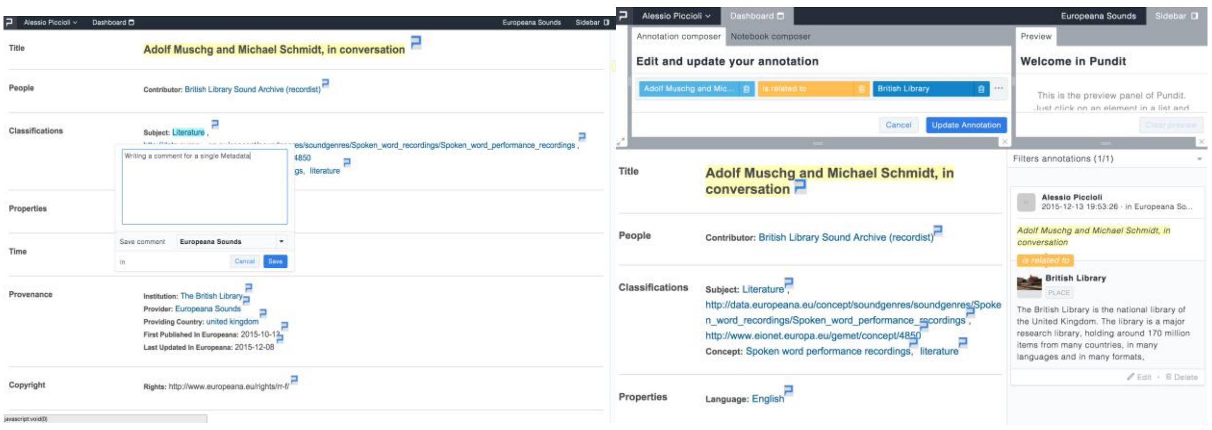
Adolf Muschg and Michael Schmidt, in conversation

Figure 32: A CHO in the content provider website and the corresponding view in the Europeana platform.



The British Library Board acknowledges the intellectual property rights of those nam
contributors to this recording and the rights of those not identified.

Figure 33: This image is a proposal for adding a button that allows the opening of the widget directly from the website of the content provider.



Title: Adolf Muschg and Michael Schmidt, in conversation
People: Contributor: British Library Sound Archive (recordist)
Classifications: Subject: Literature , …es/soundgenres/Spoken_word_recordings/Spoken_word_performance_recordings , …4850 …gs, literature
Writing a comment for a single Metadata
Properties
Time
Provenance: Institution: The British Library
Provider: Europeana Sounds
Providing Country: united kingdom
First Published In Europeana: 2015-10-12
Last Updated In Europeana: 2015-12-08
Copyright: Rights: http://www.europeana.eu/rights/rr-f/

Edit and update your annotation

Welcome in Pundit

This is the preview panel of Pundit.
Just click on an element in a list and

Title: Adolf Muschg and Michael Schmidt, in conversation
People: Contributor: British Library Sound Archive (recordist)
Classifications: Subject: Literature, http://data.europeana.eu/concept/soundgenres/soundgenres/Spoken_word_recordings/Spoken_word_performance_recordings , http://www.eionet.europa.eu/gemet/concept/4850
Concept: Spoken word performance recordings, literature
Properties: Language: English

Alessio Piccioli
2015-12-13 19:53:26 · in Europeana So...
Adolf Muschg and Michael Schmidt, in conversation
is related to
British Library
The British Library is the national library of the United Kingdom. The library is a major research library, holding around 170 million items from many countries, in many languages and in many formats.

Figure 34: New widget with metadata and annotations

Left: The new layout of the widget that displays all metadata of a resource that is suitable for enrichment. Annotations referring to the whole CHO resource are made on the resource title.

Right: the widget with all the components shown (dashboards, sidebar, triple composer).

6.3 Syncing annotation to Europeana

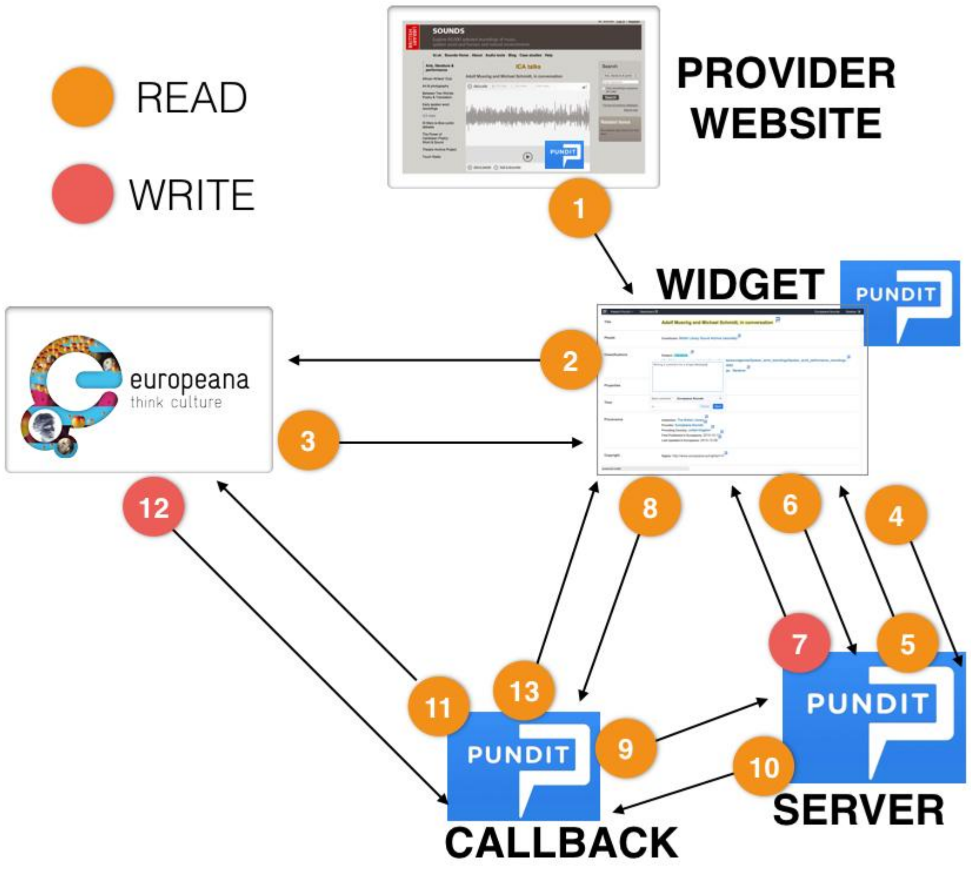


Figure 35: This image represents all steps of the annotations creation flow in the scenario where Pundit is used, and the subsequent communication (writing) to the Europeana server.

Here we explain in details all the steps:

1. Content Provider Websites has a button that opens-the Pundit widget in a new window. The unique parameter used is the CHO ID in the Europeana Server.
2. The widget calls the Europeana API in order to retrieve the CHO metadata.
3. Europeana API sends back to the widget the metadata values and the widget renders the annotation interface with Pundit client instantiated. Every metadata that can be annotated will have the Pundit ICON that starts the annotation user flow.
4. The Pundit client sends to the Pundit Server the information needed to retrieve all the annotations already existing connected to the CHO
5. The Pundit server sends back all the annotations already existing and the Pundit client renders them in the widget
6. As soon as the user finishes and saves the annotation, the Pundit client sends the annotation data to the Pundit server
7. The Pundit server creates an annotation and sends back to the client the new annotation ID (unique value in the Pundit Server)
8. When the annotation is saved and the new ID is sent back to the client, the client calls a callback

script (POSTSAVE) that manages the new annotation data flow from Pundit server to Europeana server. The client sends to the callback the new annotation ID

9. The callback calls the pundit server sending it the new annotation ID. This step is necessary (is not possible to do it directly from the annotation created to the client) because when the annotation is created the server add to the annotation some metadata needed for the mapping dedicated to the writing process to Europeana
10. The Pundit server sends back to the callback the new annotation data. The callback elaborates the annotation and it prepares the data such that it can be received by the Annotation API component.
11. The callback sends to the Europeana Annotation API component the new annotation data
12. The Europeana Annotation API component creates the new annotation and, in case of success, it sends back to the callback a 201 HTTP signal
13. The callback sends to the Pundit client the corresponding acknowledgment signal and the user receive a successful message

The previously described process can also be run asynchronously (through the Round Tripping Daemon). When the callback receives a new annotation from the client, it does all the preparation for communication (step 10) and instead of calling directly saves data prepared in a queue that can be cleared later with the appropriate calls to the server or Europeana by the server itself. The queue is emptied when the saving operation of the annotation server Europeana is successful.

6.4 Roadmap for the Enrichment Annotation Scenario

The enrichment annotation scenario foresees the following functions (user stories)[65]:

Table 4: Roadmap for Annotations

User story	Epic	Description	Prio	Deadline
18	3	Annotate a CHO	MH	January, 2016
18	3	Semantically tag a CHO	MH	January, 2016
18	3	Link a CHO to another	MH	January, 2016
1	1	Annotate CHO metadata / Suggest a new statement to an existing CHO description (E1)	NTH	July, 2016
9	12	Annotate CHO metadata / Suggest a change to an existing statement of a CHO description	MH	July, 2016
9	5	Annotating an existing Annotation / Reply to a previous CHO annotation (UC11.A, S9/E5)	MH	July, 2016
10	5	Annotating an existing Annotation / Commenting a previous CHO annotation (UC11.B, S10/E5)	MH	July, 2016
	E5	Annotating an existing Annotation / Using like/dislike (UC10.B, E9)	MH	July, 2016

[65] Refer to D2.2, MS11 and the following working spreadsheet document: https://docs.google.com/spreadsheets/d/11Rh0aR3eikc_A-k1R0nVp3WW80dL4T_J4IRFpq_7cdU/edit#gid=1667815154

7 References

Ref 1	D2.2 - Functional design of semantic enrichment http://pro.europeana.eu/files/Europeana_Professional/Projects/Project_list/Europeana_Sounds/Deliverables/EuropeanaSounds-D2.2-Functional-design-of-semantic-enrichment-v1.0.pdf
Ref 2	https://docs.google.com/document/d/1-JSL9078OWFbz8Vfnou3YC9rwDkqWywXmynBsvKBI2Y/edit
Ref 3	D2.4 Crowdsourcing infrastructure V1 Assessment and Recommendations http://pro.europeana.eu/files/Europeana_Professional/Projects/Project_list/Europeana_Sounds/Deliverables/EuropeanaSounds-D2.4-Crowdsourcing-infrastructure-V1-assessment-and-reccommendations-v1.0.pdf

Appendix A: Terminology

A project glossary is provided at: http://pro.europeana.eu/web/guest/glossary.

Additional terms are defined below:

Term	Definition
APEX	Archives Portal Europe network of excellence
CHO	Cultural Heritage Organisation
EC-GA	Grant Agreement (including Annex I, the Description of Work) signed with the European Commission
GA	General Assembly
PC	Project Coordinator
TEL	The European Library
UAP	User Advisory Panel
WP	Work Package