

# **OPEN\_NEXT**

## **Deliverable 3.4** **Populated Database**



This project is funded by the European Union’s Horizon 2020 Research and Innovation Programme under the Grand Agreement no. 869984.

## OPEN\_NEXT – Transforming collaborative product creation

Consortium:

#	Participant Legal Name	Short Name	Country
1	TECHNISCHE UNIVERSITÄT BERLIN	TUB	DE
2	INSTITUT POLYTECHNIQUE DE GRENOBLE	GINP	FR
3	ALEXANDER VON HUMBOLDT-INSTITUT FÜR INTERNET UND GESELLSCHAFT GMBH	HIIG	DE
4	UNIVERSITY OF BATH	UBA	UK
5	ZENTRUM FÜR SOZIALE INNOVATION GMBH	ZSI	AT
6	FRAUNHOFER GESELLSCHAFT ZUR FÖRDERUNG DER ANGEWANDTEN FORSCHUNG E.V.	FHG	DE
7	DANSK DESIGN CENTER APS	DDC	DK
8	WIKIMEDIA DEUTSCHLAND - GESELLSCHAFT ZUR FÖRDERUNG FREIEN WISSENS EV	WMDE	DE
9	WIKIFACTORY EUROPE SL	WIF	ES
10	STICHTING WAAG SOCIETY	WAAG	NL
11	MAKER	MAK	DK
12	AGILE HEAP EV	FLB	DE
13	SONO MOTORS GMBH	SOM	DE
14	OPNTEC GMBH	OPT	DE
15	STYKKA APS	STY	DK
16	TILL WOLFER	XYZC	DE
17	FICTION FACTORY	FIF	NL
18	M2M4ALL	SOD	NL
19	INNOC ÖSTERREICHISCHE GESELLSCHAFT FÜR INNOVATIVE COMPUTERWISSENSCHAFTEN	HAL	AT

Duration: 09/2019-08/2022

Grant: H2020-869984

Contact (coordinator): Prof Dr-Ing Roland JOCHEM

Address: Technische Universität Berlin, Sekretariat PTZ 3, Pascalstr. 8-9, 10587 Berlin

E-mail: [roland.jochem@tu-berlin.de](mailto:roland.jochem@tu-berlin.de)

**Disclaimer:** The contents of this document are not intended to replace consultation of any applicable legal sources or the necessary advice of a legal expert, where appropriate. All information in this document is provided "as is" and no guarantee or warranty is given that the information is fit for any particular purpose. The user, therefore, uses the information at his/her sole risk and liability. For the avoidance of all doubts, the European Commission has no liability in respect of this document, which is merely representing the views of the author(s).

## D3.4 – “ Populated Database”

Review and approval status:

	<b>Name and Surname</b>	<b>Role in the Project</b>	<b>Partner</b>
<b>Author(s)</b>	Dr. Linda-Rabea Heyden Elena Aleynikova	Project manager, Program Manager	WMDE
	Martin Häuer	Work Package Lead, Research Associate	FHG
<b>Reviewed by</b>	Robert Mies	Project Coordinator	TUB
	Pen-Yuan Hsing	Postdoctoral Researcher	UBA
<b>Approved by</b>	Robert Mies	Project Coordinator	TUB

History of changes:

<b>Version</b>	<b>Date</b>	<b>Description of changes</b>	<b>By</b>
0.1	15.02.2022	Initial draft	Martin Häuer, Elena Aleynikova, Dr. Linda-Rabea Heyden
0.2	28.03.2022	Second draft version	Dr. Linda-Rabea Heyden, Martin Häuer, Elena Aleynikova
0.3	05.04.2022	Draft for internal review	Dr. Linda-Rabea Heyden, Martin Häuer, Elena Aleynikova, Conny Kawohl, Hanna Petruschat
0.4	14.04.2022	Draft for external review	Dr. Linda-Rabea Heyden, Conny Kawohl, Elena Aleynikova, Martin Häuer, Hanna Petruschat
1.0	27.04.2022	Final document for final project management verification	Dr. Linda-Rabea Heyden, Elena Aleynikova, Martin Häuer

Details:

<b>Dissemination level</b>	Open Access
<b>Due date</b>	29.04.2022
<b>Issue date</b>	27.04.2022
<b>Contract No.</b>	869984
<b>Responsible Partner</b>	WMDE
<b>File name</b>	D3.4_Populated database

**Keywords:** Open source hardware, Krawler, LOSH, frontend, open source development, distributed database, linked open data, collaborative production, documentation

## Table of contents

<b>List of abbreviations and terms</b>	<b>6</b>
<b>List of Figures</b>	<b>7</b>
<b>Executive Summary</b>	<b>8</b>
<b>T3.4 Detailed Report</b>	<b>110</b>
1 Introduction	110
1.2 Background and Problem Statement	121
2 Outline of T3.4	132
2.1 Adjustment of T3.4	143
2.1.1 Timeline	143
2.1.2 Budget	154
2.1.3 Responsibilities	154
2.1.4 LOSH Frontend	154
2.1.5 Sub-Task Definition	154
2.1.6 User Testing Activities	165
3 WikibaseReconcileEdit – The Reconciliation API	176
3.1 Problem Statement	176
3.2 Technical Requirements	176
3.2.1 Testing of the API	176
3.2.2 Constraints for Certain Entity Types	187
3.2.3 Error Messages	187
3.2.4 Documentation	187
3.3 Summary	197
4 Krawler	19
4.1 Introduction	19
4.2 Krawler design notes	210
4.2.1 Fetcher	221
4.2.1 RDF-Converter	232
4.2.2 Wikibase-Pusher	232
4.2.3 Note on technical requirements	243
4.2.4 Note on Data Quality	243
4.3 Summary and Outlook	254
5 Further LOSH-Modules	265
5.1 LOSH Appropedia Scraper	265
5.1.1 Short Description and Design Notes	265
5.1.2 Usage and Installation	265
5.1.3 Summary and Outlook	276
5.2 OSH Repository Checker	276
5.2.1 Short description & design notes	276

5.2.2 Usage and Installation	298
5.2.3 Summary and Outlook	29
5.3 OKH-LOSH Manifest File Creator	29
5.3.1 Short description and Design Notes	29
5.3.2 Usage and Installation	310
5.3.1 Summary and Outlook	321
5.4 LOSH Reporter	331
5.4.1 Short Description	331
5.4.2 Design notes and usage	342
5.4.3 Summary and Outlook	364
6 LOSH-Frontend	376
6.1 Problem Statement	376
6.2 Scope and Design Specifications	387
6.3 Methodology	39
6.3.1 Workshops	410
6.4 Testing	410
6.4.1 Heuristic evaluation	410
6.4.1.1 Filter Options	421
6.4.1.2 Unclear System Status	432
6.4.1.3 Match between System and Real World	443
6.4.2 Usability Testing by Userlutions	443
6.4.2.1 First impression	454
6.4.2.2 User Guidance	465
6.4.2.3 Features	465
6.4.2.4 Profile	465
6.5 Summary	465
7 Summary and Outlook	477
Bibliography	49
8 Annex	510
8.1 Further Information on the Krawler	510
8.1.1 Data Sources of the Krawler	510
8.1.1.1 Requirements	510
8.1.1.2 Results	521
8.2 Setup and Usage of the Krawler	543
8.2.1 Requirements	543
8.2.2 Installation Instructions	543
8.2.3 Execution	554
8.2.3.1 Data collection (Fetcher)	554
8.2.3.2 Convert data into Linked Open Data (RDF-Converter)	565

8.2.3.3 Data upload (Wikibase-Pusher)	565
8.2 Additional Findings from the Heuristic Evaluation	565
8.3 Full Report on Usability Testing by Userlutions	565

## List of abbreviations and terms

API	Application Programming Interface
BDD	Behaviour Driven Development
CAD	Computer Aided Design
CTA	Call to Action
D3.4	Deliverable for T3.4
FHG	Fraunhofer-Gesellschaft (task participant)
FOSS	Free <sup>1</sup> /Open Source Software
GA	Grant Agreement
LOD	Linked Open Data
LOSH	Library of Open Source Hardware
OKH	Open Knowhow
ON	OPEN!NEXT
OSH	Open Source Hardware
PO	Project Officer
REST API	Representational State Transfer Application Programming Interface
SME	Small and Medium-Sized Enterprises
SPARQL	Simple Protocol and RDF Query Language
T3.4	Task 3.4, part of WP3
TUB	Technische Universität Berlin

<sup>1</sup> The term “free” refers to “freedom” and does not, as often misinterpreted, mean that “freely licensed” material cannot be commercially exploited.

UX	User Experience
WIF	Wikifactory (task participant)
WMDE	Wikimedia Deutschland (task leader)
WP	Work Package

## List of Figures

Fig. 0: Overall data flow of the LOSH indicating on the right hand side the data collection by the Krawler, the reconciliation by the API and population of the Wikibase, and the user facing LOSH frontend at the bottom to display the data, p. 11

Fig. 1: Modular structure of the Krawler and its interfaces to other platforms, p. 21

Fig. 2: Example issue on GitLab with assessment report directly copied from the tool's output, p. 27

Fig. 3: Rendered view of the example issue shown in Figure 2, p. 27

Fig. 4: View of the interface of the OKH-LOSH Manifest File Creator while editing an uploaded manifest file, p. 30

Fig. 6: Detail view of a highlighted mandatory field, p. 32

Fig. 7: Sample from the OSHdata report 2020 showing the distribution of hardware licences among OSHWA-certified products, p. 33

Fig. 8: Schematic data flow within the LOSH Reporter, p. 34

Fig. 9: Sample from the LOSH Report 04/2022 showing the distribution of crawled OSH designs from different OSH online platforms, p. 37

Fig. 10: Losh-Frontend Website and Modern Design, p. 38

Fig. 11: LOSH Frontend Architecture, p. 38

Fig. 12: Data Explorer with Search box and filters, p. 39

Fig. 13: Instructions for data submission, p. 39

Fig. 14: “Organisation” filter allowing selection by scrolling and clicking, but designed with a lighter grey hue making it appear disabled, p. 42

Fig. 15: While loading a “no data” image is shown, p. 42

Fig. 16: Items do not display version number, but Git commit hash, p. 43

Fig. 17: Items with empty name fields, p. 43

Fig. 18: Registration on Wikifactory.com, p. 44

Fig. 19: View of the website with the small icon for the chat option in the bottom right corner, p. 44





## Executive Summary

This document aims to present the progress on the OPEN!NEXT<sup>2</sup> Wikibase instance for a wide range of Open Source Hardware<sup>3</sup> (OSH) products named Library of Open Source Hardware (LOSH). The enhancement of the knowledge base as described in the grant agreement entails its population with OSH data via two main channels: a Reconciliation Application Programming Interface (Reconciliation API), called WikibaseReconcileEdit, and a crawler, called Krawler.

The **Reconciliation API** consolidates, deduplicates and allows for single as well as batch input of OSH data into the Wikibase instance. It guarantees a quality minimum thanks to a quality check prior to data submission to the Wikibase. For example, it features restricting constraints for certain entity types that show helpful descriptive error messages during the data input as a warning for invalid scope parameters: “X is not a valid value, must be one of...”. It also checks if the OSH and the data for input are available under a free/open licence.

The **Krawler** is a software module within the LOSH system, in regular intervals collecting OSH metadata from priorly defined OSH online platforms and features the principle data source for LOSH. It only collects data about OSH modules available under officially endorsed free/open licences that can be exploited by the community. Its automatic regular retrieval of new OSH designs keeps the knowledge base up to date.

Further **sub-modules** enhance the OPEN!NEXT (ON) knowledge base by increasing the available metadata to be retrieved by the Krawler (LOSH Appropedia Scraper), by assessing the quality of OSH projects in regards to community best practices on technical documentation (OSH Repository Checker), by allowing to create, manipulate or fix plain text files containing metadata about OSH designs (OSH Manifest File Creator), and by automatically extracting statistical data from the LOSH knowledge base in form of reports (LOSH Reporter).

All modules ensure that the API is filled with useful and discoverable data that is complete and up to date. They will ensure the growth of the knowledge base beyond the project’s end.

The Wikibase instance does not feature a user friendly interface as is. A fact that was not taken into consideration during the grant proposal and now corrected with the development of the **LOSH-Frontend**. The frontend makes LOSH data accessible in a user reactive, styled webpage, using a modern frontend framework. It also guides makers to submit their open source hardware data. The frontend was tested twofold in a) a heuristic evaluation and b) user tests conducted by the subcontracted agency Userlutions.

As expected, the Wikimedia Deutschland (WMDE) testing of the LOSH-Frontend revealed several problems of which the following three need further attention in our opinion:

the following three major problems need further attention:

1. The filtering for exploring data gets users stuck in seemingly unusable states.
2. Parts of the content still contain placeholder text.
3. Some bundle downloads do not work or demand a login.

These problems were analysed by WMDE and FHG and could be addressed in the T3.5.

The user tests conducted by Userlution for user journeys on the Wikifactory website revealed that users are missing clear guidance due to a small chat- or unusual menu-icon or clear instructions like “Import project here..”.

<sup>2</sup> Abbreviated from here on as ON.

<sup>3</sup> Abbreviated from here on as OSH.

The recommendations on how to eliminate these problems were developed and handed over to the site's owner WIF for future implementation.

The Reconciliation API (WikibaseReconcileEdit), the LOSH-Frontend and the Krawler (the crawler for the OPEN!NEXT project) constitute three channels to access and populate the database. The database gives access to the documentation and designs and offers a means for the community to connect as well as build further relationships in the network as more SMEs and grassroot communities are invited to this ecosystem.

The [Wikibase instance for OSH metadata](https://losh.ose-germany.de/)<sup>4</sup> is published and publicly available with its LOSH frontend [here](https://losh.opennext.eu/).<sup>5</sup> It is the foundation for the massive database on OSH products and services and offers an advanced query service.

It is currently populated with 31 706 items that reference over 22 380 OSH objects, events, and services.<sup>6</sup> This exceeds the projected population of 1000 referenced OSH products and services. Because of the population by crawler, the number is rising constantly.

The LOSH has several advantages over other online available databases of OSH solutions, as it is neither limited to a specific field of technology, nor bound to prior certification or onboarding processes. As the ON Wikibase instance only stores data about OSH modules available under officially endorsed free/open licences, reuse and commercial exploitability are fully sanctioned.

The LOSH frontend, the default language settings of the Wikibase instance, and the documentation for all parts is in English and therefore available for international usage.

Requirements for the system modules were mainly based on user stories from the former deliverable of the Task 3.1. (T3.1) and work initiated in Task 3.3 (T3.3).

The task T3.4 is a joint effort of the partners Wikimedia Deutschland e.V. (WMDE), Fraunhofer IPK (FHG), and Wikifactory (WIF) with WMDE leading the task. The original concept of the task needed adjustments in regards to its timeline, budget allocation, assigned responsibilities and most importantly the addition of the LOSH-Frontend application.

The T3.4 development work was carried out by WMDE and FHG mainly, with WIF supplying work on their API. The task and deliverable D3.4 was finalised in February 2022. The enhancement of the demonstrators is scheduled for the upcoming T3.5.

---

<sup>4</sup> <https://losh.ose-germany.de/>

<sup>5</sup> <https://losh.opennext.eu/>

<sup>6</sup> Retrieved on 22.03.2022.

## T3.4 Detailed Report

### 1 Introduction

This task is the last step for developing software in the WP3, this software helps to initiate data collecting and move forward automatically data population. The software-development related tasks in the WP3 included the following steps for the platform creation:

- T3.1 aimed to analyse user scenarios and collect needs for collaborative engineering;
- T3.2 focused on design of features and data ontology and development of standards for the future OSH ecosystem;
- T3.3 initiated development of the platform which was based on a Wikibase software, provided by WMDE, Wikifactory data, and OSH server provided by TUB.
- T3.4 focused on development and testing of access points to data, such as a frontend for LOSH and Wikifactory platform, as well as development of tools for data population such as Krawler and Reconciliation API.

Once all these steps are finished, the ON has a software platform which will be additionally validated in T3.5.

The delivered software in the frame of the T3.4 is open source and the code is possible to find and continue to work on on the GitHub platform which has a big technical-oriented community. The software doesn't require installation on a user's computer and is fully available online.

The data flow as it is represented in figure 0 was presented in the D3.3 and demonstrates the overall model of the platform and its components interacting.

With the LOSH frontend added and finished in T3.4 the visualisation was finished with the added component. As can be seen in the graphic on the top right hand side, the Krawler retrieves data via APIs that is then reconciled by the WikibaseReconcileEdit API to populate the database. The stored data can then be accessed by the users (shown in the bottom left corner) via the LOSH frontend.

The LOSH has several advantages over other online available databases of OSH solutions, such as [OHO.wiki](https://oho.wiki). LOSH is neither limited to a specific field of technology, as e.g. CERN's [OHWR.org](https://ohwr.org), nor bound to prior certification or onboarding processes, such as the registry of [OSHWA-certified OSH](https://oshwa-certified.org). Most importantly, the ON Wikibase instance only stores data about OSH modules available under officially endorsed free/open licences, ensuring e.g. full permissions for reuse and commercial exploitability.

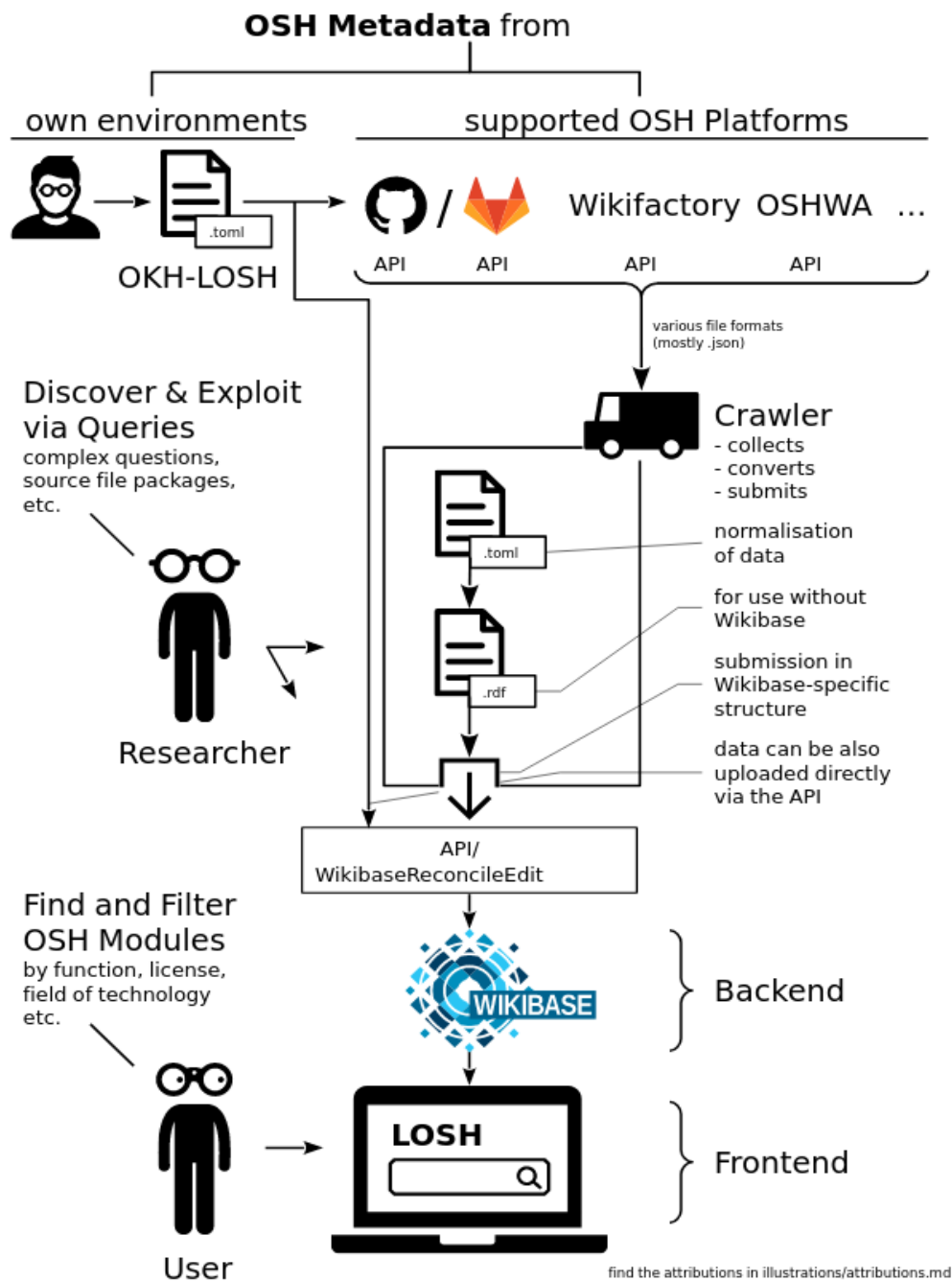


Figure 0: Overall data flow of the LOSH indicating on the right hand side the data collection by the Krawler, the reconciliation by the API and population of the Wikibase, and the user facing LOSH frontend at the bottom to display the data.

## 1.2 Background and Problem Statement

Currently, data about OSH projects and services is scattered across various platforms and not yet linked in a centralised OSH-data hub that makes all open source LOD findable in one place.

The ON metadatabase aims to be this place. Since the Ecosystem as it was planned required an entry point to the data collection, it was necessary to build a special repository for storing data on the one hand and on the other hand a frontend to this repository for making the data available for the OSH users.

For the repository the Wikibase software was set-up in T3.3. Wikibase is a suite of knowledge-base software for managing linked-open data, which was initially developed for the [Wikidata](https://www.wikidata.org/wiki/Wikidata:Main_Page)<sup>7</sup> project. Wikidata, a free and open knowledge base for structured data, is an example of a Wikibase instance. Since Wikidata was introduced to the world, many institutions, organisations, libraries, archives and museums set up their own Wikibase instance and so implemented Wikibase in order to open their linked data to the world.

Wikibase suite is an open source software which is designed to collect data in a machine-readable form that can be read and edited by humans and machines alike. Wikibase allows users to set up an instance with a data model that is flexible with keeping data in different languages and in a number of formats including JSON, RDF/XML, N3 and YAML. It also offers an opportunity to query data using a SPARQL endpoint. This choice of software was perfectly fitting to the open source ideology and possible future tasks of the OSH community, as it helps to work on data in a collaborative and linked way without disadvantages of proprietary software. This software choice helped to solve the problem of creating a repository for the distributed database of LOSH. But because querying the Wikibase requires knowledge of SPARQL, it was decided to add a frontend to make the stored data available to users without SPARQL proficiency.

For the purpose of the ON project and T3.4, the Wikibase instance was installed and set up in order to host OSH-related data in T3.3. To populate the database that was only modelled but did not contain any data yet, two channels were devised. The first is a reconciliation API and the second the Krawler. Both work to populate the Wikibase. The API allows submission via two endpoints for single and batch entities. The Krawler automatically retrieves relevant data.

## 2 Outline of T3.4

The task T3.4 consists of the following subtasks: the development of an API to transfer data from WIF and other websites into a common data repository. Additionally, data collection by a web crawler, augmented by workshops, usability testing, and frontend development (giving a user friendly surface to explore the data repository).

Most architecturing and engineering was done by WMDE with the other partners FHG and WIF participating. The data transfer from WIF and other websites was split between WMDE and WIF as well as FHG with a role in linking to websites other than WIF to the repository. In addition, a frontend was developed and tested. Two types of usability tests were conducted by WMDE. One internally and one that was delegated to an external contractor with 50 participants.

---

<sup>7</sup> [https://www.wikidata.org/wiki/Wikidata:Main\\_Page](https://www.wikidata.org/wiki/Wikidata:Main_Page)

Continuous development of respective demonstrators was done by FHG and WIF.

Partner	Sub-task
<i>WMDE (lead)</i>	<ul style="list-style-type: none"> <li>➤ Develop an API or interface to transfer data from WIF to the repository</li> <li>➤ Frontend developing (new)</li> <li>➤ Conducting usability tests</li> <li>➤ Participation in a data collection workshop</li> </ul>
<i>FHG</i>	<ul style="list-style-type: none"> <li>➤ Data collection</li> <li>➤ OSH-Repository-Checker</li> <li>➤ Open Knowhow-LOSH (OKH-LOSH) Manifest File Creator</li> <li>➤ LOSH Appropedia Scraper</li> <li>➤ LOSH Reporter</li> <li>➤ Continuous development of respective demonstrators</li> </ul>
<i>WIF</i>	<ul style="list-style-type: none"> <li>➤ Support in developing an API or interface to transfer data from WIF to the repository</li> <li>➤ Participation in the UX testing</li> <li>➤ Continuous development of respective demonstrators</li> </ul>

## 2.1 Adjustment of T3.4

From its initial conception, the task had to be adjusted. Changes included the timeline, budget, responsibilities and detailing of sub-tasks, and user testing activities. Through the adjustments, the partners were able to accomplish the aspired goals for T3.4 and the deliverable D3.4 of a populated database.

### 2.1.1 Timeline

To accomplish the task, an extension of the initial deadline was requested and approved by the Project Officer.

The extension was necessary mostly to accommodate for the delayed T3.3. The previous task required additional frontend development for the data visualisation of the Wikibase instance. This influenced the start and work on T3.4.

As part of the delays, the initial subcontracting for the crawler was insufficient to achieve the required goals and an additional subcontractor was required. Unexpected unavailability of the principal Krawler developer led to further delays.

Due to postponed implementation of functionalities, the respective testing activities of these functionalities were subsequently delayed as well.

### 2.1.2 Budget

A reallocation of budgets from personnel to other costs was requested and approved by the Project Officer. WMDE used the reallocated budget to hire a User Experience (UX) agency which was responsible for testing Wikifactory's deliverables of the Work Package 3 (WP3). Outsourcing the testing was necessary in order to keep the timeline and provide testing with a considerable number of participants for the user tests.

In addition, a freelance developer was hired for the frontend development to react to feedback from the first iterations.

### 2.1.3 Responsibilities

The development of the Krawler was shifted from WMDE to FHG before the initial phase of the project in 2019, contrary to the GA that states development by WMDE.

When the necessity of the LOSH frontend became apparent, it was initially a FHG responsibility. But due to engineering and UX expertise and resources, the responsibility for the frontend was shifted to WMDE.

As already outlined in the D3.3 report, the Wikibase, WikibaseReconcileEdit (API), and the LOSH frontend are already hosted by OSEGeV, contrary to the grant agreement that states FHG hosting it on TUB servers. The decision was made during T3.3. and described in D3.3. Hosting by OSEGeV provided the opportunity to host LOSH for 3 years minimum at their own costs after the ON was finished, it was also eliminating an extra effort of moving software from TUB servers to OSEGeV servers. However, hosting the Wikibase instance by a third party led to additional coordination efforts and has halted development as well as fixing issues.

### 2.1.4 LOSH Frontend

Already during work on D3.3 it became clear that the interface to transfer data to the Wikibase would need an additional interface. Since Wikibase's generic frontend was insufficient for data exploration in our specific use case, a more user appropriate frontend was required to make it more accessible for users without software skills or proficiency of Simple Protocol and RDF Query Language (SPARQL). This frontend for LOSH was designed, developed, tested, and subsequently improved during T3.4.

### 2.1.5 Sub-Task Definition

During task T3.3 its was defined that the following tools should be developed to make the ecosystem workable and stable:

- LOSH Appropedia Scraper
- OSH Repository Checker
- OKH-LOSH Manifest File Creator
- LOSH Reporter

While the definition of these modules was started in T3.3, the finalisation happened in T3.4.



### 2.1.6 User Testing Activities

During the course of the project, it was agreed that the Hackathon, mentioned in the GA, was replaced with user research activities. The hackathon format itself was deemed not to be the optimal approach to produce valid user experience results and the amount of planned 500 participants was excessive for this purpose. Instead, a focused user research approach was chosen with overall 50 participants. These research activities were done in cooperation with an external UX testing agency which provided professional support. They hired and interviewed relevant groups of users, analysed the results and provided recommendations for the improvement of the workflows, as well as user interfaces in the WP3.



## 3 WikibaseReconcileEdit – The Reconciliation API

### 3.1 Problem Statement

The Library of Open Source Hardware (LOSH) should be filled with meaningful data that is easily discoverable, complete and up to date. The WikibaseReconcileEdit<sup>8</sup> acts as a reconciliation tool that ensures accuracy and data quality of any OSH data submitted to the Wikibase instance.

### 3.2 Technical Requirements

The purpose of the Reconciliation API is to consolidate, deduplicate and allow for single as well as batch input (through an alternative endpoint) of OSH data into the Wikibase instance.

Statements on an item support reconciliation of not only the wikibase-item type, but also support reconciliation against other databases' items. In addition, the API includes enforcing constraints for certain entity types that give helpful descriptive error messages in order to make the data input more user friendly. To increase the user experience even further, reconciliation was extended to properties and labels, instead of IDs for the specifications of properties, and the documentation was extended.

To ensure support by the open source community, and adhere to standards of other free/open source projects, the WikibaseReconcileEdit was published on [Github](#)<sup>9</sup> to increase its public reach and availability. It is licensed under the GNU General Public Licence v2.0.<sup>10</sup>

This strategic decision influenced the final architecture and meant an adjustment of the initial version, so that the Reconciliation API would not only work with Wikifactory as a use case, but has an overall broader application for data reconciliation. The main adjustments were another endpoint for submitting batches of entities at a time and of making it possible to specify properties by their label rather than their property. Both were fully accomplished.

#### 3.2.1 Testing of the API

For initial testing, WMDE set up a test system<sup>11</sup> and created some initial properties that map to the OPEN!NEXT ontology. The test system was a “lighter” version of the Wikibase instance developed for OPEN!NEXT. It allowed for testing of the functionality of data input, as well as the interaction with the crawler. This test system was only used in the beginning and we moved gradually to the actual Wikibase to develop the Reconciliation API.

A special page was used to make the development on the API more comfortable for engineers. To use the reconciliation functionality, the page included one box for entity parameters (JSON)

---

<sup>8</sup> <https://github.com/wmde/WikibaseReconcileEdit>

<sup>9</sup> <https://github.com/wmde/WikibaseReconcileEdit#readme>

<sup>10</sup> <https://github.com/wmde/WikibaseReconcileEdit/blob/main/COPYING>

<sup>11</sup> [https://wikibase-reconcile-testing.wmcloud.org/wiki/Main\\_Page](https://wikibase-reconcile-testing.wmcloud.org/wiki/Main_Page)

and another box for reconcile parameters (JSON), as well as a button to submit to reconciliation services.

The initial idea was that the page should have polled the single reconciliation endpoint instead of the endpoint for batch reconciliation, and handled authentication internally.

Redirects to the newly created item after successful submission and displaying error messages in the form were also not implemented in the end. The WikibaseReconcileEdit was successfully developed, implemented and tested with all intended functionality.

### 3.2.2 Constraints for Certain Entity Types

In general, Wikibase offers different data entity types (e.g. Item, Property, Lexeme, Form, Sense). Sometimes properties are used differently in the entity types. It was therefore necessary to add a functionality to allow restricting constraints to certain entity types.<sup>12</sup>

Restricting the entity type to which a constraint applies to is important in order to avoid false constraint violations, because a property is sometimes used in different contexts in the different entity types. To keep the maintenance script to import constraint entities working, it is designed to allow variables to default to the same property. It also means that no constraint violation should be triggered when the restriction applies to the edited entity type. In case the added value is invalid, a warning message is displayed in the form of “X is not a valid value, must be one of...”.

### 3.2.3 Error Messages

When saving the status in the single edit endpoint and the batch edit endpoint, error messages are necessary. So errors that come from saveStatus for the single and for the batch edit endpoint were attached to the response. Descriptive error messages for all API endpoints are provided to help users of the API help debug issues with batch editing.

### 3.2.4 Documentation

We worked on making the documentation approachable for all users of the Wikibase Reconciliation API. For example, the documentation around working with some of the Wikibase caches<sup>13</sup> was further improved.

---

<sup>12</sup> Cf. the phabricator ticket <https://phabricator.wikimedia.org/T269724>. Phabricator is the collaboration platform used by WMDE for managing work in software projects. Generally, it is open to all Wikimedia contributors. It can be found [here](#). The phabricator board for the ON project can be found [here](#).

<sup>13</sup> <https://phabricator.wikimedia.org/T285653>

### 3.3 Summary

The Wikibase Reconciliation API is a general purpose module that allows us not only to batch import datasets in the specific use case of ON and LOSH, but also in more general terms for all kinds of Wikibases if they choose to use the module for their purposes. It has been tested intensively and provides approachable documentation for immediate usage by a wide set of users. It continues to work as the main conductor of data from sources such as the Krawler into the LOSH Wikibase instance and is thereby an integral part of the LOSH ecosystem of applications.

## 4 Krawler

### 4.1 Introduction

The Krawler is a software module within the LOSH system collecting OSH metadata from priorly defined OSH online platforms and features the principle data source for LOSH. Depending on the public Application Programming Interface (API) an online platform provides, the Krawler can collect data by either of the three following ways:

1. directly from the API of the OSH online platform (fully integrated platform),
2. using manifest files found using the API of the OSH online platform (connected platform),
3. using additional custom scripts and manifest files (associated platform),

However, the latter should only be used as a temporary or bridging solution. Writing custom scripts require significant extra effort in development (compared to the other ways) and are usually subject to dependencies on website and content structure, which may change over time.

The Krawler only collects data about OSH modules available under officially endorsed free/open licences, ensuring e.g. full permissions for reuse and commercial exploitability. Through periodical execution of the Krawler the knowledge base is kept up to date and new OSH designs are added automatically. This keeps maintenance at a minimum – in contrast to fully manually curated OSH registries such as from the [OKH project](#). As data is just mirrored from the source, OSH developers and platform owners keep the full sovereignty over their data. A dedicated module of the Krawler (the RDF-Converter) transforms the crawled content into Linked Open Data (LOD), qualifying the knowledge base to 5 from 5 stars according to Tim Berners-Lee’s deployment scheme for open data.<sup>14</sup>

By now, LOSH features the largest free knowledge base of audited OSH designs from different platforms available online. For details about the selection of data sources and obstacles found on the way please see chapter 8.1 in the annex.

The Krawler has been designed for a server-based application allowing for automatic, periodic execution, but can also be executed locally by any willing party<sup>15</sup>. Due to its modular nature and free/open source code it can be easily extended, copied or adapted to more use cases. The addition of more data sources or data collection methods is especially easy, since only a new “fetcher” module needs to be added (see the Krawler design notes below).

The source code has been documented and published under a free/open licence through the following GitHub-repository: <https://github.com/OPEN-NEXT/LOSH-Krawler/>

---

<sup>14</sup> Matheus, Ricardo, et al. “Open Government Data and the Data Usage for Improvement of Public Services in the Rio De Janeiro City.” *Proceedings of the 8th International Conference on Theory and Practice of Electronic Governance*, 2014, <https://doi.org/10.1145/2691195.2691240>.

<sup>15</sup> Since the full execution includes the upload to a Wikibase instance that has the WikibaseReconcileEdit module installed, this last step would require access to such an instance. If no such instance is defined, data is stored locally only.

Please find details regarding setup and usage of the Krawler either in the linked GitHub repository or in the annex in chapter 8.1.

## 4.2 Krawler design notes

The Krawler as a whole applies an ETL cycle (**E**xtract, **T**ransform and **L**oad data):

1. copy data from defined, heterogeneous sources (OSH online platforms) after defined criteria,
2. convert collected data into LOD using the OKH-LOSH specification and ontology to make it available for further purposes such as querying and analyses,
3. upload the converted data to defined data stores.

This reflects in the software structure of the Krawler: Each of its modules can be executed independently on a bash level. Modules either connect to defined APIs using deposited credentials (Fetcher) or work with locally stored files (RDF-Converter) or both (Wikibase-Pusher). By its current state, consists of three modules:

- Fetcher (for data collection),
- RDF-Converter (for data conversion into Linked Open Data (LOD)),
- Wikibase-Pusher (for data upload to the target Wikibase instance), which are outlined in the following subchapters. Figure 1 illustrates the data flow between these modules.

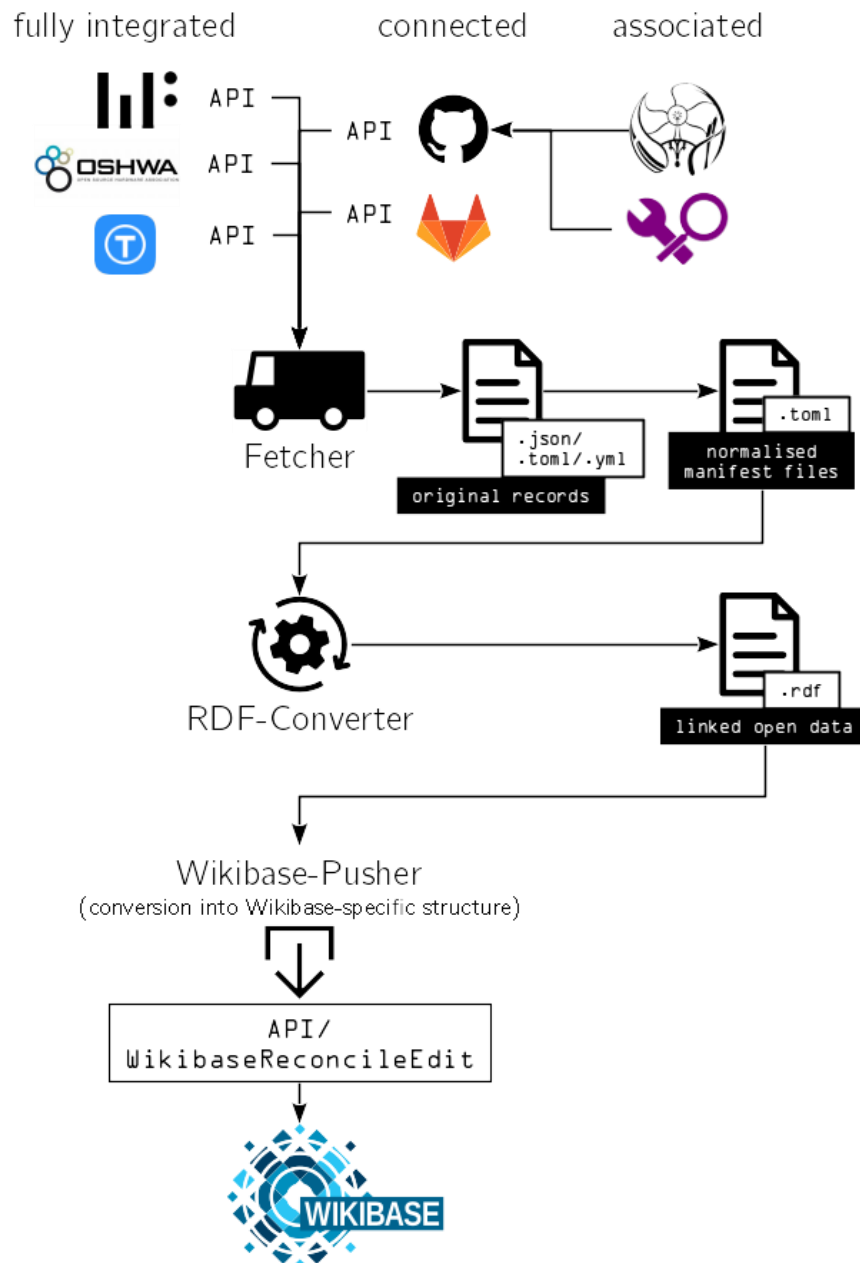


Figure 1: Modular structure of the Krawler and its interfaces to other platforms

#### 4.2.1 Fetcher

The Fetcher extracts OSH metadata from the targeted source systems (see Figure 1 above). Since every source system applies its own data structure and also different formats (YAML, TOML, JSON etc.) this results in a heterogeneous data set. During the extraction, the Fetcher filters the projects, keeping only the ones with non-empty repositories and an officially endorsed free/open licence (see chapter 8 in the annexure for details). The original record of eventually extracted data is stored locally on the system from which the Fetcher has been executed. This is particularly useful to assess and validate the data quality from different sources or for the analysis of edge cases. As a practical example, several Wikifactory projects

with ambiguous licensing terms have been found in one of these analyses<sup>16</sup>. It was found that the ambiguity was provoked by a faulty configuration file on the Wikifactory platform, which then has been fixed by the team. In a last step, the original record is then converted into a normalised data set for further processing by the RDF-Converter.

The Fetcher itself consists of submodules for each data source, currently:

- github.com (including all manifest files that have been collected/created using scripts),
- gitlab.com,
- opensourceecolog.gitlab.com,
- wikifactory.com,
- certification.oshwa.org,
- thingiverse.com.

All extracted data is timestamped, allowing for e.g. historical analyses.<sup>17</sup>

#### 4.2.1 RDF-Converter

In the data transformation stage, the normalised data set (manifest files in TOML format) from the Fetcher is converted into RDF using the OKH-LOSH ontology – by which yet isolated objects and properties become connected as LOD. The RDF output is stored separately on the system from which the RDF-Converter has been executed. In the case of LOSH, OSEGeV, which hosts the system, publishes all RDF outputs in a versioned format in a dedicated public GitLab repository<sup>18</sup> under a permissive free/open licence for further exploitation. At this point, data can be already queried and analysed if loaded into any RDF framework (e.g. Apache Jena).

#### 4.2.2 Wikibase-Pusher

In the load phase, data is uploaded to the end target(s) where the data shall be stored. While various end targets could be defined for this stage, the upload will need to meet the data structure expected by the targeted data store. Consequently, as Wikibase requires a special input format, this module is limited to push data only to Wikibase instances, expecting that the WikibaseReconcileEdit module is set up there. The latter takes care of Wikibase-internal mapping to the specific structure Wikibase uses to store its data.

Data is pushed as crawled, meaning that any reconciliation with the target data store must be conducted *at* the target data store<sup>19</sup>. Hence it is subject to the decision of the target how and which data shall be processed. This policy was applied since for external data stores (potentially the Open Hardware Observatory or the Open-KnowHow Search) these requirements would be initially unknown, also eliminating the possibility of independent modifications of this reconciliation for the target data store.

<sup>16</sup> E.g. projects stating “CERN OHL” as their hardware licence, whereby five different licences containing this designation are being distributed by CERN, some with significantly different licensing schemes (e.g. CERN-OHL-S-2.0 as a strongly reciprocal licence, CERN-OHL-P-2.0 as a permissive licence).

<sup>17</sup> As applied for the LOSH Reporter, cf. ch. 5.4.

<sup>18</sup> <https://gitlab.opensourceecology.de/verein/projekte/losh-rdf> (CC0-1.0)

<sup>19</sup> In the case of LOSH this is done by the WikibaseReconcileEdit module for Wikibase and git (computing the diffs) for the RDF data set on GitLab.

#### 4.2.3 Note on technical requirements

The described structure keeps the Krawler usable within any RDF framework. Furthermore modules can be used independently e.g. if only raw metadata from OSH platforms is interesting, bare execution of the fetcher suffices.

Software has been developed following the Unix philosophy<sup>20</sup>; specifically:

- a strictly modular approach has been applied and
- output of one software module is expected as input for another (even unknown) module.

Consequently, outputs from the Krawler modules have been designed to be easily processable for external software modules. In fact, the LOSH Reporter (cf. ch. 5.4) could be seen as an example case for this principle since it acts as a external module, taking the RDF data from the official GitLab repository<sup>21</sup> as any willing party could, too.

This practice keeps code complexity low, also lowering the barrier for external contributors to modify it. As a result the Krawler is easy to run (or host), test, customise, maintain and develop further. This was key requirement in development since the tool is meant to be further maintained and developed by the OSH community<sup>22</sup>. As a side effect of this practice, software modules become applicable to a large variety of use cases including many not envisioned by the original developers.

#### 4.2.4 Note on Data Quality

As outlined, the quality and informative value of crawled data significantly depends on its source. Prior work (precisely in the task T3.3 “Development of the metadatabase using Wikibase instance”) led to the creation of a metadata specification that enabled the procession of data from various OSH online platforms by bridging their data structures. However, since the specification has only recently been published, it has not been widely adopted, yet.<sup>23</sup> To provide rich metadata fully complying with the OKH-LOSH specification, manifest files have been deemed to be the only suitable solution within the scope of this work package.

However, Wikifactory joined the development of the OKH-LOSH specification in an early stage and became the very first OSH online platform adopting it in their API. Through the implementation of GraphQL, their API offers a feature-rich option to query the Wikifactory platform independently from LOSH. For the Krawler, the Wikifactory platform is consequently the only OSH online platform by now that is a) fully integrated and b) fully compliant with the OKH-LOSH specification. The result is a large and rich dataset that is available not only on the Frontend (cf. ch.6 ), but also for automated statistical reports on OSH metadata (cf. ch. 5.4). As such, the case of Wikifactory can serve as an example for other OSH online platforms.

---

<sup>20</sup> Which is a set of cultural norms and philosophical approaches to minimalist, modular software development, based on the experience of leading developers of the Unix operating system and widely adopted in the FOSS ecosystem.

<sup>21</sup> <https://gitlab.opensourceecology.de/verein/projekte/losh-rdf>

<sup>22</sup> most likely in the Open-KnowHow ecosystem

<sup>23</sup> Broad adoption is aimed to realise by cooperating with the OKH initiative which is committed to maintain and promote the specification beyond the OPEN!NEXT project.



### 4.3 Summary and Outlook

The design approach chosen for the development of the Krawler fully covers the scope for this work package and shows an overall good performance in practice. By 26th March 2022 over 59,000 OSH projects had been collected across nine different sources, while data extraction is still ongoing, mainly limited by query restrictions from APIs. The total number of crawled OSH projects is expected to exceed 1 million within the next three months.<sup>24</sup>

The Unix philosophy applied in the development of the Krawler led to a modular software solution that can be easily adopted, extended and exploited for a variety of (also commercial) use cases. Furthermore, required maintenance is kept at a minimum to facilitate the further hosting by open source communities.

LOSH-related tools, including the Krawler have been presented in various public community workshops.

OSEGeV, as the community-based hoster of the LOSH system, will provide this service until the end of the ON projects plus for a minimum of three years after. OSEGeV and the OKH initiative committed to the further development of the LOSH system<sup>25</sup>, so tools, data and specification will be further improved and maintained in a community-based process beyond ON. To support this, FHG will continue to participate in the OKH maintenance group until the end of the project.

The joint data collection, processing and publication inside LOSH, enabled by the Krawler, practically features the first bridge between OSH online platforms which (presumably) haven't been in contact before. Data profiling, based on the outputs of the Krawler, may be a good start to discuss specifications for a common data structure and to generally improve data quality for the OSH community.

---

<sup>24</sup> Most of these projects would come from OSHA and Thingiverse. The Thingiverse API in particular drastically limits the speed of the initial data extraction. Once all projects have been visited once, the Fetcher revisits only qualified projects, checking them for updates, and focuses on new uploads.

<sup>25</sup> For instance, as data extraction takes time, the three phases of the ETL process could be executed in pipeline. Furthermore, resource consumption, enhanced error logging (also for faulty data) and improved documentation have been raised as concerns for the further development.

## 5 Further LOSH-Modules

### 5.1 LOSH Appropedia Scraper

#### 5.1.1 Short Description and Design Notes

The LOSH Appropedia Scraper features a tool to extract OSH metadata from the Appropedia platform using manifest files, making its data available for LOSH. [Appropedia](https://www.appropedia.org/Appropedia:About) is an online platform to develop and share collaborative solutions in sustainability, poverty reduction and international development through appropriate technology and research.<sup>26</sup> It has been chosen as an essential community and platform partner, as being a co-developer of the OKHv1 specification and the first OSH platform adopting the specification. A test crawl using the tool on December 15th 2021 returned 1847 OSH projects; find the data attached to this report in the “Appropedia Test Crawl” ZIP container in the subfolder “raw data”.

Thanks to a fruitful cooperation with open source developers from the Appropedia community, the tool could be significantly simplified during the development process. Instead of, as when the development was started, scraping raw source code from wiki articles (that contain the technical documentation of OSH projects there) it now fetches manifest files directly, which are auto-generated on Appropedia’s side.<sup>27</sup> As a consequence, the tool does not depend anymore on the data structure inside source code of wiki articles – which could break easily.

Extracted data is stored in two different files:

- [appro\\_proj\\_names.csv](#) - the list of project names
- [appro\\_yaml\\_urls.csv](#) - the table containing the OKH v1 file URLs (same format as used by the OKH initiative for the Open-KnowHow Search Platform)

Data extraction is automatically executed weekly, updating these files.

The source code has been documented and published under a free/open licence through the following GitHub-repository: <https://github.com/OPEN-NEXT/LOSH-Appropedia-Scraper>

#### 5.1.2 Usage and Installation

This command-line tool is executable from any platform (Windows, mac or Linux) and neither additional proprietary software nor admin rights to function. It can be used on local machines (e.g. for private or research purposes) or integrated into automated toolchains on servers. Consequently downloading the files from the public GitHub repository<sup>28</sup> will suffice, no subsequent configuration required. Users can then run the tool on the command line simply by executing it:

```
./appro-fetcher
```

---

<sup>26</sup> <https://www.appropedia.org/Appropedia:About>

<sup>27</sup> on <https://openknowhow.appropedia.org/manifests/>

<sup>28</sup> <https://github.com/OPEN-NEXT/LOSH-Appropedia-Scraper> (Unlicense)

Results are exported in CSV format and appear in the `target` subdirectory.

However, the tool has been primarily designed to be automatically executed weekly on GitHub itself (using the “build-bot”); results are hosted on the pages of the same repository.<sup>29</sup> Hence, for its primary use case, no manual interaction is required. This also applies when users fork the project; as long as it’s hosted on GitHub, it will continue to work fully automatic.

### 5.1.3 Summary and Outlook

The tool features a first practical case for a data source that was made available for the Krawler using dedicated scripts and is a stable element within the currently deployed productive system of LOSH. As a fully-automated tool running on GitHub services, it requires neither manual interaction nor maintenance<sup>30</sup> in order to fulfil its purpose.

The development of this tool resulted in a best case scenario of how collaboration between LOSH and OSH platforms can work in the future. The team reached out to Appropedia representatives (concretely the current executive director Emilio Velis<sup>31</sup>) and discussed joint efforts. In the subsequent collaboration, the quality for metadata was improved, also resulting in an improved data quality on the LOSH side. Furthermore, Appropedia decided to develop a new tool for the platform, which is now providing users with on-demand automatically generated manifest files, easing data extraction also beyond LOSH.

Currently, the community collaboration proceeds in order to further improve the quality of metadata and to increase the quantity of crawlable OSH content on the Appropedia platform.

## 5.2 OSH Repository Checker

### 5.2.1 Short description & design notes

The OSH Repository checker is a command line tool for quality assessment of OSH projects (“technical project linting”). It checks whether a project repository adheres to community best practices regarding how technical documentation is stored in git repositories (which have been identified to be a top choice for OSH development).<sup>32</sup>

The tool analyses locally stored repositories and returns an assessment report. The report uses the markdown format so that it can be pasted directly into the issue description of the corresponding OSH project. The output renders as a ToDo list for each yet open point to be resolved for full compliance with community best practices – see Figure 2 and 3.

---

<sup>29</sup> Data extraction happens weekly at 03:02 each Monday, UTC; the latest results can be downloaded under the following URL: [https://open-next.github.io/LOSH-Appropedia-Scraper/appro\\_yaml\\_urls.csv](https://open-next.github.io/LOSH-Appropedia-Scraper/appro_yaml_urls.csv)

<sup>30</sup> apart from bug fixes in case e.g. dependencies break or interfaces change

<sup>31</sup> <https://www.appropedia.org/User:Emilio>

<sup>32</sup> Mies, R., et al. “Development Of Open Source Hardware In Online Communities: Investigating Requirements For Groupware.” *Proceedings of the Design Society: DESIGN Conference*, vol. 1, 2020, pp. 997–1006., [doi:10.1017/dsd.2020.38](https://doi.org/10.1017/dsd.2020.38).

## New Issue

**Title** technical project linting results

Add [description templates](#) to help your contributors to communicate effectively!

**Type** Issue ?

**Description**

Write Preview

**B I ” </> @**

Hey there, thanks for this awesome project! I did a quick check using the [OSH Tool] (<https://gitlab.com/OSGermany/osh-tool>) from the [OPEN!NEXT project] (<https://opennext.eu/>) and found a few issues, find them listed below. Let me know if I can be helpful in order to resolve these :)

- [x] LICENSE exists
- [ ] OKH file exists
  - HEAVY - Open Know-How meta-data file ([okh.toml](#)) not found. Please consider creating it. See <[https://github.com/OPEN-NEXT/OKH-LOSH/blob/master/sample\\_data/okh-TEMPLATE.toml](https://github.com/OPEN-NEXT/OKH-LOSH/blob/master/sample_data/okh-TEMPLATE.toml)> for a template.
- [x] README exists
- [x] Clean electronics files
- [x] Might be generated
- [ ] Clean CAD files
  - LIGHT - File-format issue(s) with 'bla.cgr': not Open, not text-based
  - LIGHT - File-format issue(s) with 'bla.CGR': not Open, not text-based
- [x] No sources in root
- [x] No unwanted files
- [x] No space in file names

Markdown and quick actions are supported [Attach a file](#)

Figure 2: Example issue on GitLab with assessment report directly copied from the tool's output

## New Issue

**Title** technical project linting results

Add [description templates](#) to help your contributors to communicate effectively!

**Type** Issue ?

**Description**

Write Preview

Hey there, thanks for this awesome project! I did a quick check using the [OSH Tool](#) from the [OPEN!NEXT project](#) and found a few issues, find them listed below. Let me know if I can be helpful in order to resolve these :)

- ☒ LICENSE exists
- ☐ OKH file exists
  - o HEAVY - Open Know-How meta-data file ([okh.toml](#)) not found. Please consider creating it. See [https://github.com/OPEN-NEXT/OKH-LOSH/blob/master/sample\\_data/okh-TEMPLATE.toml](https://github.com/OPEN-NEXT/OKH-LOSH/blob/master/sample_data/okh-TEMPLATE.toml) for a template.
- ☒ README exists
- ☒ Clean electronics files
- ☒ Might be generated
- ☐ Clean CAD files
  - o LIGHT - File-format issue(s) with 'bla.cgr': not Open, not text-based
  - o LIGHT - File-format issue(s) with 'bla.CGR': not Open, not text-based
- ☒ No sources in root
- ☒ No unwanted files
- ☒ No space in file names

Figure 3: Rendered view of the example issue shown in Figure 2

Adhering to such practices has several benefits for different user groups when dealing with the technical documentation of the project, e.g.:

- higher familiarity for users/replicators, external contributors, manufacturers and reviewers (e.g. from conformity assessment bodies according to DIN SPEC 3105-2);

- improved efficiency for git-based services (smaller, easier to handle repositories, meaningful diffs);
- any software dealing with the content can recognise project metadata and essential files (e.g. source files) much easier;
- content can be easier ported to other hosting platforms (or any other system dealing with projects) (less lock-in effect);
- improved comparability among OSH projects;
- improved reusability e.g. when used as a module within another OSH project (e.g. an open source phone using an open source GPS module).

The source code has been documented and published under a free/open licence through the following GitLab-repository: <https://gitlab.com/OSGermany/osh-tool>

### 5.2.2 Usage and Installation

The tool is executable on a bash level, e.g. locally the user's machine. Consequently downloading the binaries from the public GitLab repository<sup>33</sup> will suffice, no subsequent configuration required. Users can then run the tool on the command line simply by executing it. To display the usage information run:

```
osh -h
```

To assess a repository, open a the command line interface in the repository's directory and run:

```
osh check
```

Results will be plotted directly in the command line interface and may look like this:

```
- [x] LICENSE exists
- [ ] OKH file exists
  - HEAVY - Open Know-How meta-data file (okh.toml) not found.
    Please consider creating it.
    See <https://github.com/OPEN-NEXT/OKH-
    LOSH/blob/master/sample_data/okh-TEMPLATE.toml> for a template.
- [x] README exists
- [x] Clean electronics files
- [x] Might be generated
- [ ] Clean CAD files
  - LIGHT - File-format issue(s) with 'bla.cgr': not Open, not
  text-based
  - LIGHT - File-format issue(s) with 'bla.CGR': not Open, not
  text-based
- [x] No sources in root
- [x] No unwanted files
```

<sup>33</sup> <https://gitlab.com/OSGermany/osh-tool> (AGPL-3.0-or-later)

- [x] No space in file names

This plot uses the markdown format, which is the standard markup language for open source projects (both software and hardware) and also used on most OSH online platforms (e.g. git-based platforms or Wikifactory).

### 5.2.3 Summary and Outlook

Throughout the research carried out in the OPEN!NEXT project and the development conducted in this work package many OSH projects have been found with incomplete documentation or to use inappropriate file formats or repository structures. As an alternative to silent acceptance of such deficiencies (without notice from the originators), the OSH Repository checker provides a very fast and easy way to a) identify deficiencies and b) report them in a constructive manner to the developers of the corresponding OSH project.

The tool has been used within this work package for a quick pre-assessment of OSH projects for the manually curated dataset (cf. ch. 8.1 in the annexure for details). It has also been presented to OSEGeV which now plans to integrate it for the pre-assessment of projects requesting a community-based review at its conformity assessment body according to DIN SPEC 3105-2. Yet these pre-assessments were carried manually, summing up to a significant effort to be run by volunteers for these reviews. Additionally OSEGeV committed to the further community-based development and maintenance of the tool. For the further development, assessments shall be extended towards a more detailed level. Furthermore, the tool shall already support fixing the encountered issues, where possible. For instance the manifest file (`okh.toml`) could be initialised with the information found in the repository.

## 5.3 OKH-LOSH Manifest File Creator

### 5.3.1 Short description and Design Notes

The OKH-LOSH Manifest File Creator offers a web interface to create, manipulate or fix manifest files.

Manifest files are plain text files containing metadata about an OSH project. For platforms that don't offer a suitable API for the Krawler (such as [GitHub](#)) manifest files are the only way to integrate OSH projects into the LOSH knowledge base. The same applies to online hosting services that don't provide any API at all (such as project-specific web pages or the [Appropedia platform](#)) – as long as the access to manifest files can be communicated to the Krawler, metadata can be extracted.

However, writing plain text metadata files can be challenging for users with no prior coding experience. Syntax errors would result in failed data parsing, while the user itself wouldn't get any feedback on whether their manifest file is corrupt as the error occurs during crawling.

The tool is configured by the official JSON schema deployed alongside with the OKH-LOSH specification.<sup>34</sup> When the specification is changed (by changing the JSON schema), the data fields in the interface of the OKH-LOSH Manifest File Creator change accordingly. Since the tool is meant to be further maintained on a community basis, it has been designed in a way to keep these maintenance efforts at a minimum.

The source code of the tool has been published under a free/open licence on the GitLab instance of OSEGeV, which also hosts all LOSH-related tools: <https://gitlab.opensourceecology.de/verein/projekte/okh-losh-manifest-creator>.

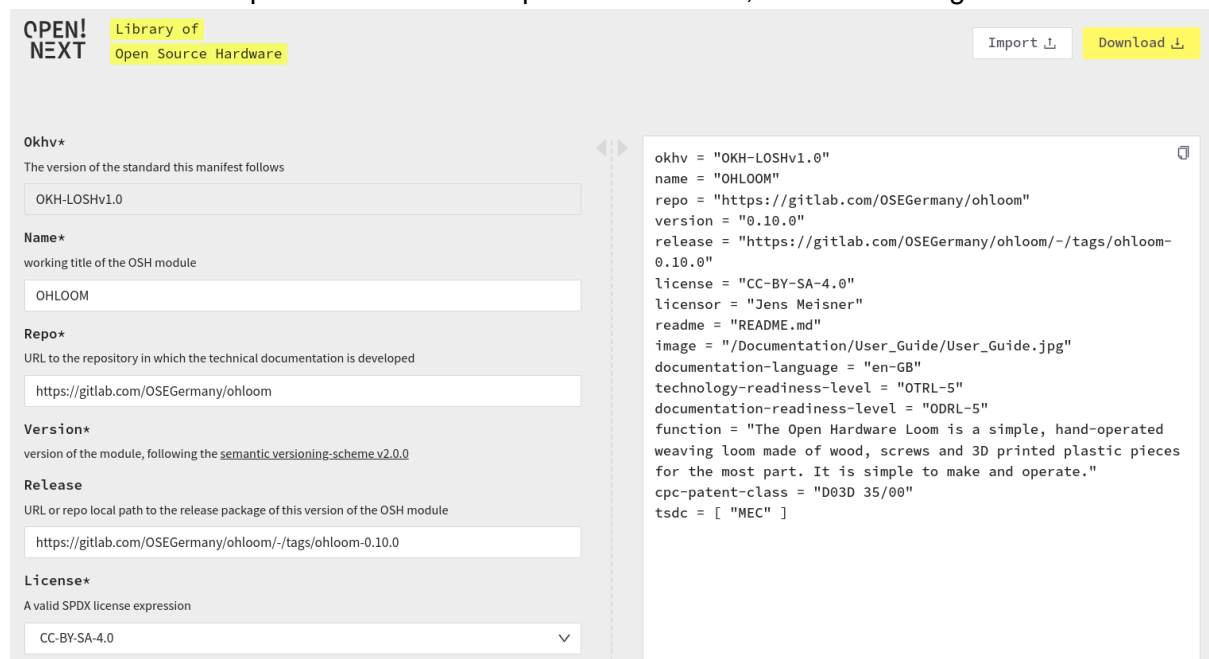
Deployment of the tool is done using GitLab’s “continuous deployment” functionality. Hence, no further installation steps are required to use the tool.

### 5.3.2 Usage and Installation

As mentioned in the previous section, no installation is required since it is deployed directly from the GitLab repository. Users only need a web browser to work with it.

With the OKH-LOSH Manifest File Creator users can create, edit and validate manifest files without needing to read and understand the OKH-LOSH specification or the required syntax for the defined data fields. The tool can be accessed under the following URL and runs in the browser: <https://manifest.opennext.eu/>

The interface is kept minimalist and is split into two sides, as shown in Figure 4.



The interface is split into two main sections. The left section contains form fields for creating or editing a manifest file, and the right section displays the resulting JSON manifest file.

**Form Fields (Left):**

- Okhv\*:** The version of the standard this manifest follows. Value: OKH-LOSHv1.0
- Name\*:** working title of the OSH module. Value: OHLOOM
- Repo\*:** URL to the repository in which the technical documentation is developed. Value: <https://gitlab.com/OSEGermany/ohloom>
- Version\*:** version of the module, following the semantic versioning-scheme v2.0.0
- Release:** URL or repo local path to the release package of this version of the OSH module. Value: <https://gitlab.com/OSEGermany/ohloom/-/tags/ohloom-0.10.0>
- License\*:** A valid SPDX license expression. Value: CC-BY-SA-4.0

**JSON Manifest File (Right):**

```
okhv = "OKH-LOSHv1.0"
name = "OHLOOM"
repo = "https://gitlab.com/OSEGermany/ohloom"
version = "0.10.0"
release = "https://gitlab.com/OSEGermany/ohloom/-/tags/ohloom-0.10.0"
license = "CC-BY-SA-4.0"
licensor = "Jens Meisner"
readme = "README.md"
image = "/Documentation/User_Guide/User_Guide.jpg"
documentation-language = "en-GB"
technology-readiness-level = "OTRL-5"
documentation-readiness-level = "ODRL-5"
function = "The Open Hardware Loom is a simple, hand-operated weaving loom made of wood, screws and 3D printed plastic pieces for the most part. It is simple to make and operate."
cpc-patent-class = "D03D 35/00"
tsdc = [ "MEC" ]
```

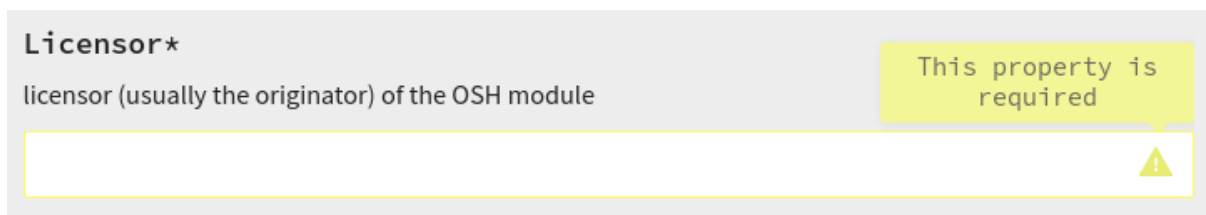
Figure 4: View of the interface of the OKH-LOSH Manifest File Creator while editing an uploaded manifest file

Manifest files can be either created directly in the web interface (using the left side of the page) or uploaded and edited. The final manifest file is shown on the right side of the page. Editing

<sup>34</sup> <https://github.com/OPEN-NEXT/LOSH-OKH-JSON-Schemas>



is allowed on both sides as they are synchronised. Consequently, users could edit or paste sections directly on the “code side” (right) and presume in the dialogue boxes on the left. Where possible, the interface offers tooltips and predefined options to choose from, e.g. a drop-down menu containing all officially endorsed free/open licences. This should additionally support users when determining metadata for their OSH project for the first time. Not yet filled, mandatory fields are highlighted, as shown in Figure 6. Consequently, the tool includes an automatic validation of manifest files to ensure that resulting files can be parsed by the Krawler without errors.



*Figure 5: Detail view of a highlighted mandatory field*

### 5.3.1 Summary and Outlook

The LOSH Manifest File Creator features an intuitive tool enabling users with low or no prior coding experience to provide rich metadata for OSH designs. As it’s a web-based tool, no further software dependencies must be pre-installed to use it, other than a modern web browser. Since the tool relies on JSON-Schema, which is shipped alongside with the official OKH-LOSH metadata specification, no further maintenance efforts are needed apart from maintaining the metadata specification itself – once it changes, the fields in the tool change accordingly.

Like the other tools in the LOSH ecosystem, the LOSH Manifest File Creator has been developed in cooperation with the OSH community, specifically the OKH initiative to ensure 1) relevance for the target group and 2) long-term maintenance. As the web interface already contains all metadata fields and explanations and takes care of the correct syntax in the output manifest file, users do not need to read the OKH-LOSH metadata specification in order to be able to submit rich metadata about their OSH design. This is specifically useful in hackathons, when people with diverse professional backgrounds may drop in and out of the event as their schedule allows it and don’t need an extensive introduction before they could start working.

Together with future versions of the OSH Repository Checker, which may be able to generate parts of a manifest file automatically, this forms part of a powerful toolset to link OSH designs to the LOSH knowledge base that would not be accessible otherwise (cf. ch. 4 for details).



## 5.4 LOSH Reporter

### 5.4.1 Short Description

The OSHdata project<sup>35</sup> was an initiative from the OSH community to give an overview of the current situation in the OSH community by crawling data from the OSHWA certification list.<sup>36</sup> The initiative published annual reports containing mostly statistical evaluations (a sample is shown in Figure 6) and distributed them in several different social media channels, reaching e.g. over 580 followers via twitter.<sup>37</sup> The last report was published for 2020;<sup>38</sup> the project was shut down thereafter due to a lack of resources. In 2021, the Open Hardware Observatory e.V. (non-profit) took over the OSHdata project in a collaboration with OPEN!NEXT.<sup>39</sup>

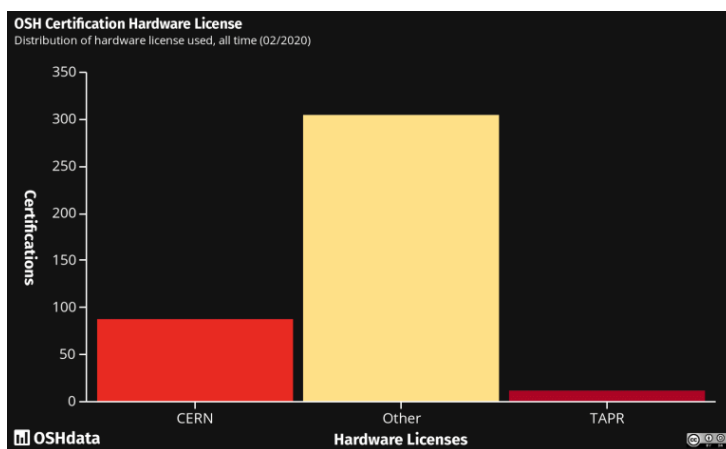


Figure 6: Sample from the OSHdata report 2020 showing the distribution of hardware licences among OSHWA-certified products

The LOSH Reporter is a command line tool with a locally running RDF environment, extracting statistical data from the LOSH knowledge base. Once set up, the tool can be executed automatically, e.g. on a server. Input of the tool is raw RDF data; as output it generates an HTML file that can be easily hosted as a web page (e.g. on [OSHdata.com](https://oshdata.com/)) and a printable PDF. The output is based on a Markdown template and uses pandoc<sup>40</sup> for exporting – consequently many more export formats, such as EPUB or DOCX would be available if needed.<sup>41</sup> Although Markdown is technically a programming language, it aims to be easy-to-use for people with no prior coding experience (e.g. GitHub uses Markdown for documents and issues, Markdown is also very similar to Wikitext, the markup language used on e.g. Wikipedia). As a result, the template, and thus the report, are fully modifiable at low effort. In

<sup>35</sup> <https://oshdata.com/>

<sup>36</sup> <https://certification.oshwa.org/list.html>

<sup>37</sup> <https://twitter.com/oshdata>

<sup>38</sup> <https://oshdata.wpcomstaging.com/2020/02/19/2020-state-of-open-hardware/>

<sup>39</sup> <https://oshdata.com/2021/07/27/new-oshdata-team/>

<sup>40</sup> Pandoc is a widely used FOSS document converter that can be easily integrated in automated workflows.

<sup>41</sup> Find a full overview of conversion options in the official pandoc documentation here: <https://pandoc.org/diagram.svg?v=20220404105907>

fact, the tool can be used for any knowledge base or dataset, although it has only been tested for LOSH yet.

#### 5.4.2 Design notes and usage

Like the Krawler, the LOSH Reporter has been developed following the UNIX philosophy (cf. ch. 4.2 for details), which resulted in a modular software structure that can be easily modified or adapted to other use cases. Figure 7 illustrates the data flow within the tool. In a first step, the tool executes a list of pre-defined queries on raw RDF data<sup>42</sup> using an RDF framework<sup>43</sup>, which outputs the desired statistical data. A Markdown template contains all the written information of the report and also defines how statistical data is meant to be rendered (e.g. where in the report certain data shall be rendered as a vertical bar chart). The document converter (pandoc) merges template and data and exports rendered versions of the report in the determined formats (currently PDF and HTML).

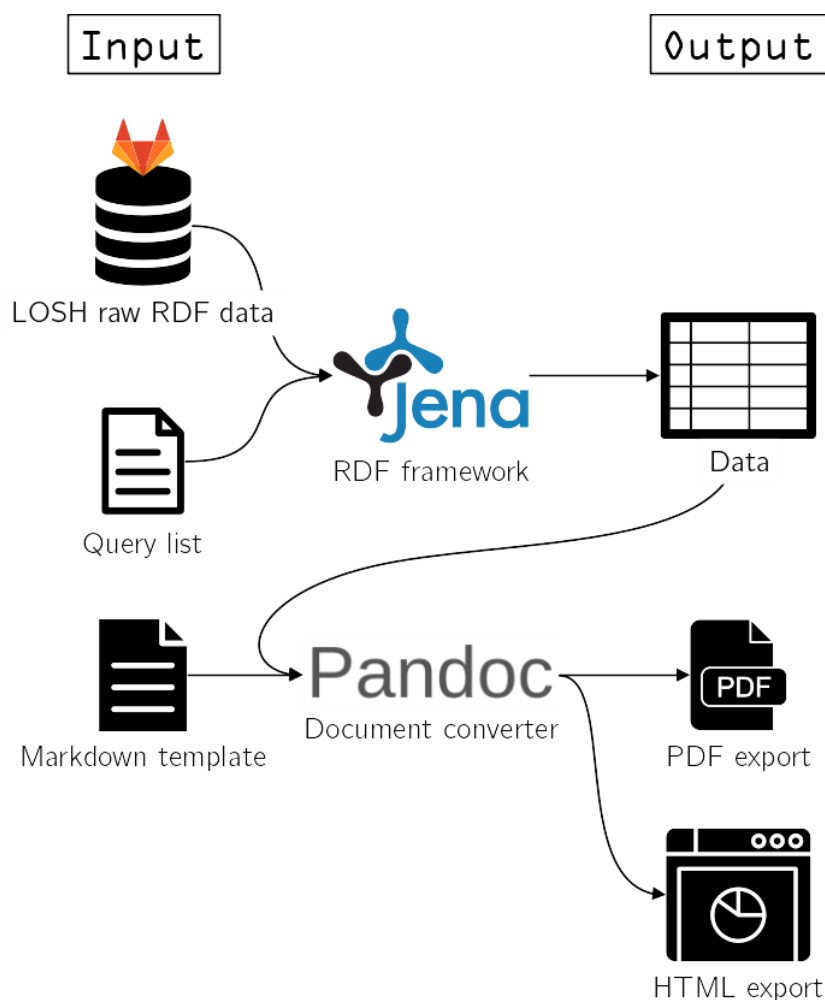


Figure 7: Schematic data flow within the LOSH Reporter

<sup>42</sup> In the case of LOSH this data is available at <https://gitlab.opensourceecology.de/verein/projekte/losh-rdf> under a free/open licence.

<sup>43</sup> concretely Apache Jena; <https://jena.apache.org/>

The tool doesn't require installation – it can just be downloaded from its GitHub repository and executed. However, it requires a few dependencies (e.g. a recent version of Python installed); for full details on dependencies, please see the README in the GitHub repository.<sup>44</sup>

To run, the tool is executed by a single command line:

```
poetry run python reporter/cli.py generate --out <OUTFOLDER> --log_to
<LOGGINGTARGET> <REPORTNAME> <format>
```

- **<format>** specifies the output format (e.g. HTML, PDF or MD)
- **<log\_to>** determines if the log is just displayed (“console”) or if it is saved in a file e.g. for troubleshooting (“file”)
- **<REPORTNAME>** is the name of the folder containing the input files needed to create a new report (and will be used as the name of the report to be generated)
- **<OUTFOLDER>** is the name of the folder where the created report will be saved.

An example command could be:

```
poetry run python reporter/cli.py --log_to console --out output generate
example html
```

The source code has been documented and published under a free/open licence through the following GitHub-repository: <https://github.com/OPEN-NEXT/LOSH-Reporter/>

Reports are published regularly under a free/open licence in the following GitHub repository: <https://gitlab.opensourceecology.de/verein/projekte/losh/losh-reports>.

Figure 8 shows a graphic from that report, giving an overview on the current dataset of the LOSH knowledge base.

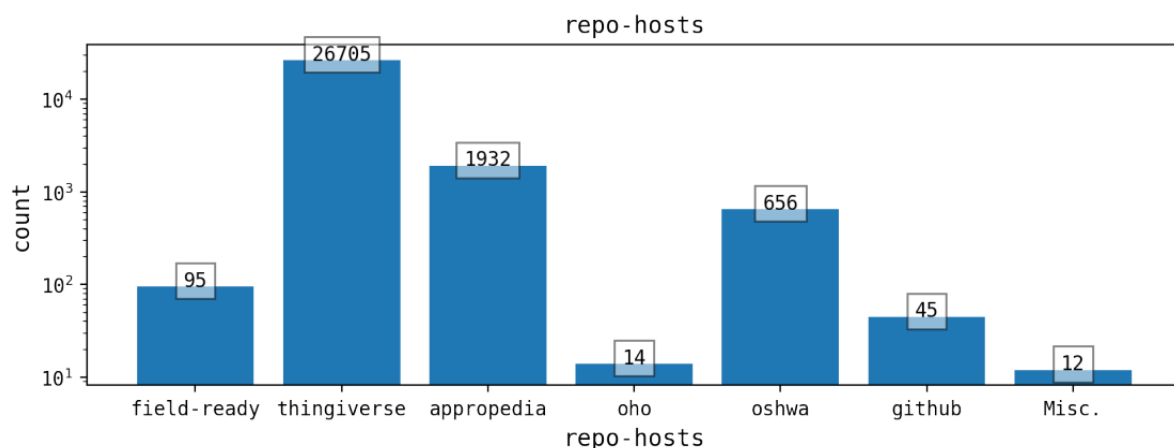


Figure 8: Sample from the LOSH Report 04/2022 showing the distribution of crawled OSH designs from different OSH online platforms

<sup>44</sup> <https://github.com/OPEN-NEXT/LOSH-Reporter/blob/main/README.md>

### 5.4.3 Summary and Outlook

Although facing a growing demand for third-party proofs of documentation quality, mostly from economic actors<sup>45</sup> and despite large public interest, the OSHdata initiative permanently stopped the publication of statistical reports on this concern – primarily because invested efforts did not compensate themselves sufficiently by a reliable business model:

*“Our research paid for itself through consulting projects and made an impact. Hundreds of new products were certified. We spoke with [Adafruit](#) and [SparkFun](#). We helped students publish papers. We [discussed open hardware in Machine Design](#). We supported [OpenForum Europe](#)’s work for the European Commission”* (OSHdata).

The LOSH Reporter is a tool fully automating (most) of the related efforts, pushing the costs for the publication of these reports to a minimum. Furthermore the content of these reports has been largely extended by processing data not only from the OSHWA Certification List (as previously done by OSHdata), but from all data sources supported by LOSH (cf. ch. 4.1 for details). The reports are now periodically published in cooperation with the Open Hardware Observatory e.V. (non-profit) which took over OSHdata in 2021.<sup>46</sup>

---

<sup>45</sup> <https://oshdata.wordpress.com/2021/04/15/shutting-down-oshdata/>

<sup>46</sup> <https://oshdata.wordpress.com/2021/07/27/new-oshdata-team/>

## 6 LOSH-Frontend

During the work on T3.3, it became apparent that a frontend is needed for the OPEN!NEXT Wikibase instance, whose generic frontend did not match the needs of our use case. A fact that regrettably was not taken into consideration during the grant proposal.

Thus, the development of the user-friendly LOSH-Frontend was added to the original concept which only states the Wikibase instance.

The frontend makes LOSH data accessible in a user reactive, styled webpage, using a modern frontend framework. It also guides makers to submit their open source hardware data. The first design iterations in T3.3 were finalised and deployed in T3.4. Accordingly, the LOSH-Frontend is now ready and available as a public [website](https://losh.opennext.eu/).<sup>47</sup>

### 6.1 Problem Statement

We based our assumptions on the following use cases:

- As a user, I would like to discover data stored in the LOSH even though I do not know SPARQL
- I would like to reuse or replicate existing OSH solutions.
- In a specific field of technology, I want to identify the relevant communities.
- I would like to submit data on my OSH products and services to make them discoverable by others in the community and SMEs.
- As SME, I would like to obtain an overview of the OSH market faster and identify gaps or overlaps in order to pilot new business opportunities.

Without the LOSH-Frontend, the data on the OPEN!NEXT Wikibase instance is only discoverable with the knowledge of SPARQL. This prevents users from finding relevant communities, scanning the OSH market, and submitting their own data on OSH products and services to be discovered by others.

LOSH is built for the community and to serve as a database about open hardware knowledge. Thanks to its frontend application design, community members who are not proficient in SPARQL can retrieve the desired information. The frontend app allows the exploration of open source hardware datasets and their projects as well as other OSH services stored in the LOSH, obtain the overview of the market, and connect to the relevant communities in a specific field of technology.

Apart from serving the community to connect, find, and contribute data on open source hardware products and services, the LOSH-Frontend allows users to contribute to the project itself by submitting issues or fixing issues themselves since the code is open source.

Thanks to its connection to the reconciliation API and the crawler that periodically retrieves OSH data from repositories, the Wikibase Instance is growing daily. Over time, the Wikibase instance on OSH will serve as a central hub for open source hardware products and services.

---

<sup>47</sup> <https://losh.opennext.eu/>

## Status Quo

So far the engagement is limited to a few partners and as the testing activities have shown, the website still has several dead ends that prohibit a full engagement and which should be addressed in the future to increase usage in the ecosystem. Ideally these issues are fixed by the community engaging in the project. The issues were analysed and ranked by severity<sup>48</sup> and possible solutions were provided. The issues should be addressed accordingly. Improvements will be made during T3.5 - validation of demonstrators.

## 6.2 Scope and Design Specifications

The LOSH-Frontend lays the basis for discovering and retrieving OSH data from the OPEN!NEXT Wikibase instance. It is a single page application which works as the interactive layer to the LOSH Wikibase instance. Its overall design was aligned with the [OPEN!NEXT website](https://opennext.eu/).<sup>49</sup>

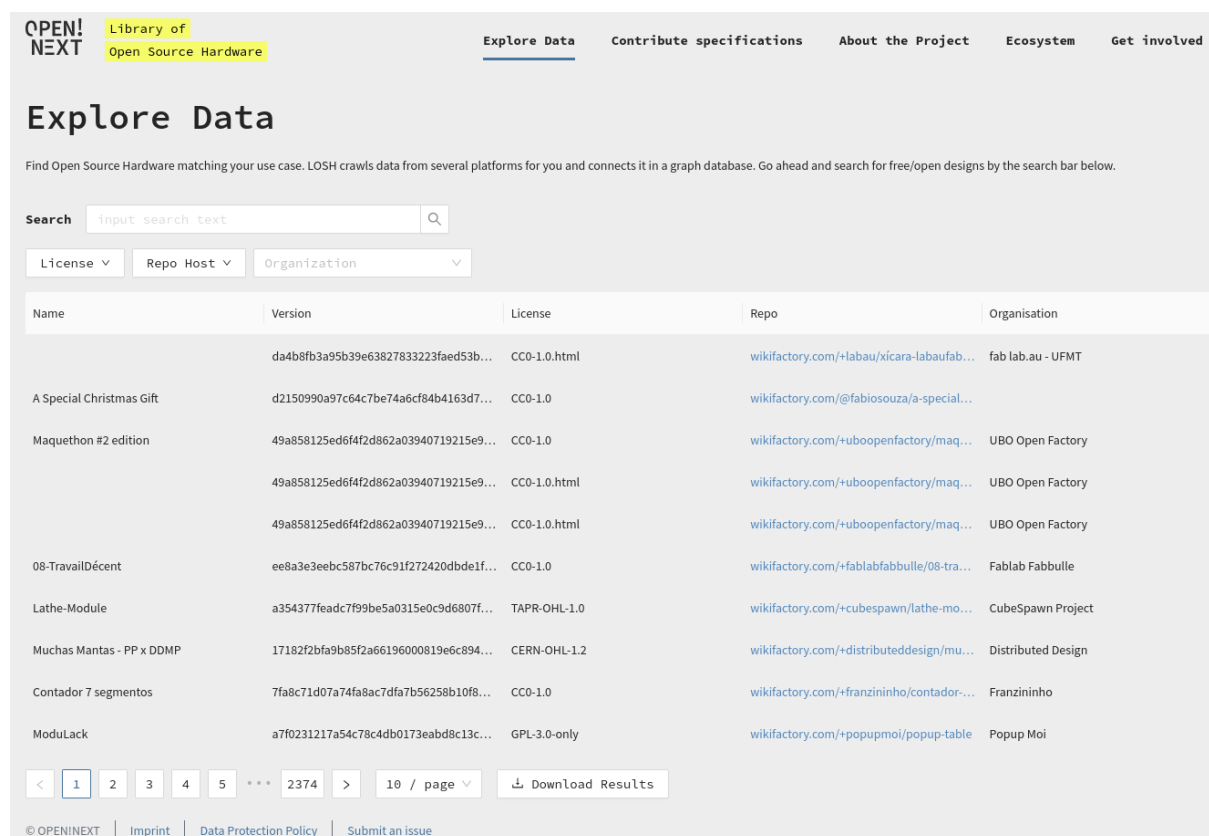


Figure 9: Losh-Frontend website and modern design

The initial specifications for the frontend were rewritten to match the adjusted requirements by the partners.

<sup>48</sup> Severity 1 being the lowest and 5 the highest.

<sup>49</sup> <https://opennext.eu/>

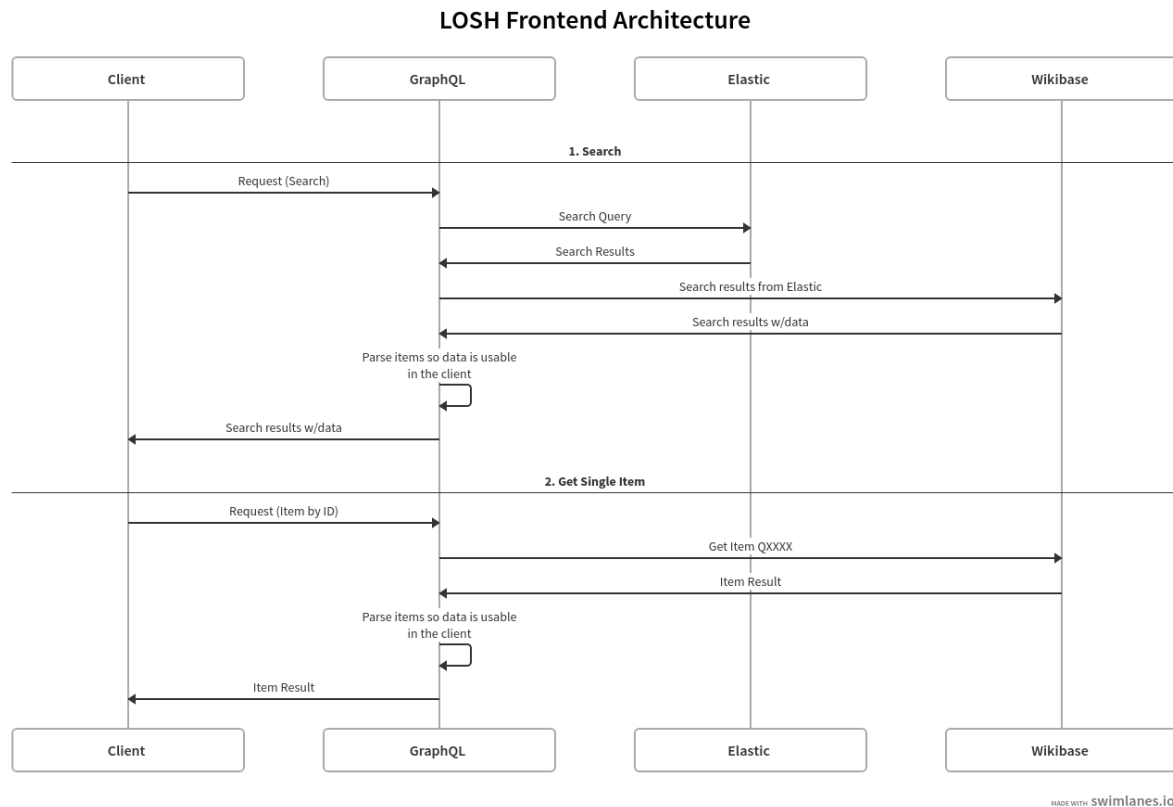


Figure 10: LOSH Frontend Architecture

The frontend contains the following structure:

- Data Explorer
- OSH Submission
- About the project
- OSH Ecosystem
- Get Involved

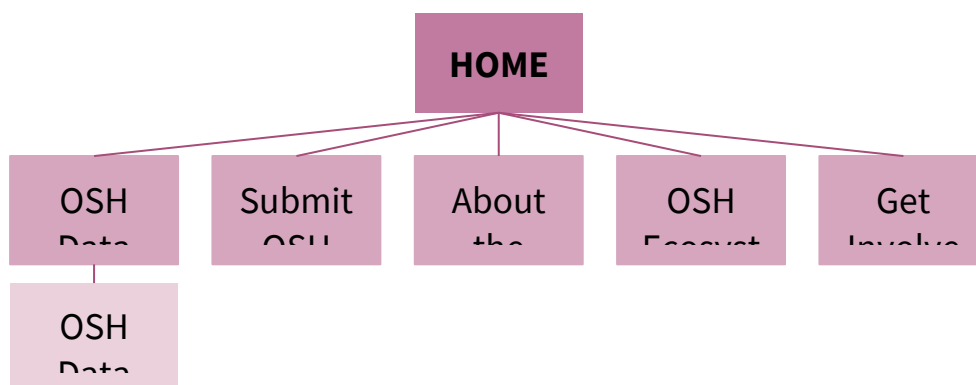


Figure 11: LOSH-Frontend Structure

### Data Explorer

To explore the data, the database can be searched via a search box. The retrieved data is displayed in a compact preview. The search results can further be refined by the parameters “Licence”, “Repo Host”, or “Organization”.

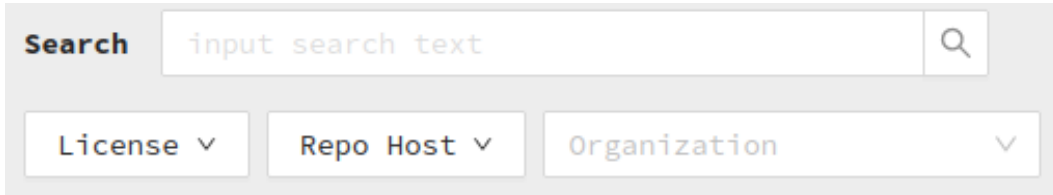


Figure 12: Data Explorer with search box and filters

The Data Explorer provides links to the submission forms for certification for the following organisations: OSHWA, OSE Germany – Conformity Assessment Body according to DIN SPEC 3105-2, OHO – Conformity Assessment Body according to DIN SPEC 3105-2.

### OSH Submission

To contribute data, a template file can be downloaded. When added to a repository via Git, the specifications will be picked up by the Krawler that goes through repositories with OSH-files periodically.

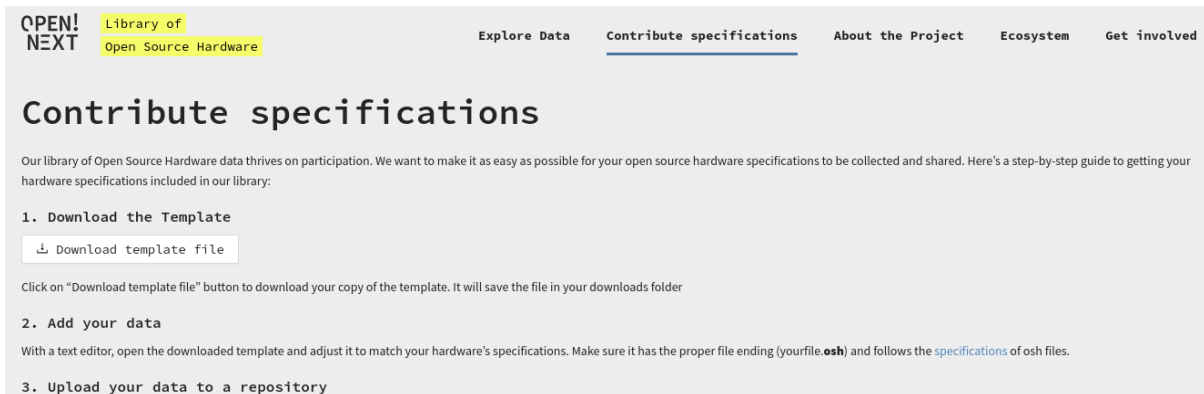


Figure 13: Instructions for data submission

### Project information

The “About the project” part provides information for everyone who wants to know more about the EU Commission funded ON project.

### Ecosystem

The “OSH Ecosystem” segment serves the purpose to link the OSH community to the other OSH related projects.

### Get Involved

With the “Get Involved” menu our users can engage open source maintainers via Github repository, report on issues and submit features requests via Github.

## 6.3 Methodology

Based on the workable wireframe and the test platform developed in T3.3, the frontend development was finished, deployed and tested in T3.4.



The team continued to apply agile development processes in working on the LOSH-Frontend. The iterative development to implement the OPEN!NEXT design style and the usability testing and evaluation happened in close collaboration with the WMDE user experience designers.

To complement the WMDE team of developers and UX designers over the course of T3.4, a freelancer was hired for the frontend development. Further involvement of developers was necessary in the end again, to quickly resolve reported code issues from the partners. The switch from the initially planned hackathon to two different ways to evaluate the user experience also led to subcontracting a user testing agency.

Like the reconciliation API, the frontend, too, was made publicly available on Github according to the ideals of FOSS. The frontend is built as a separate software module that can be automatically deployed utilising [GitHub Actions](#) or [GitLab's CI/CD](#) onto a given webspace. It offers key functionalities in regards to data discovery even for users who are not familiar with SPARQL.

### 6.3.1 Workshops

An internal workshop was conducted with WMDE in the lead and the other task partners participating, as stated in the GA. During the workshop, FHG presented five products and five workflows with our UX team. The UX team introduced the partners to several evaluation methods, showcasing pros and cons of each method to help the partners decide on useful evaluations practices.

Subsequently, WMDE decided to hire an external agency giving them the task to test products and workflows with a defined group of people. The agency Userlutions was moderating the process, working with a defined group of people, collecting and analysing data and feedback. Their report details suggestions about improving the workflows like entering data and further products in the future.<sup>50</sup>

## 6.4 Testing

To evaluate the user interface of the LOSH as well as the Wikifactory repository, two types of methods were employed. WMDE delegated usability testings to the external contractor [Userlutions](#)<sup>51</sup> and internally opted for the method of a heuristic evaluation. Full reports of the testing results can be found in the annex.

### 6.4.1 Heuristic evaluation

A Heuristic Evaluation is a process where someone trained in usability principles reviews an application: this person compares the application against a set of guidelines called Heuristics that tend to make for more usable experiences.

---

<sup>50</sup> Please see the paragraph below on user testing for further details.

<sup>51</sup> <https://userlutions.com/>

The heuristic method performs well in finding problems. It finds (partly) different problems than a user test and can find classes of problems that user tests usually can't: For example, it can uncover issues of interface inconsistency which are hard to find with a usability test.<sup>52</sup> It also has been shown to be a very cost effective evaluation method measured by time investment by experts to use the method.<sup>53</sup>

An effective Heuristic Evaluation is performed by multiple evaluators and not just one, because different evaluators tend to find different problems.

We used the well established Nielsen Guidelines in our Analysis.<sup>54</sup>

Two UX experts evaluated the LOSH UI according to the 10 Nielsen guidelines for Heuristic Evaluation of Interfaces. A usage scenario was used which covered the core use cases for the product. The tested scenario was as follows:

1. go to [losh.opennext.eu](https://losh.opennext.eu) in a web browser (desktop or mobile client)
2. type the technology of interest into the search bar (default: "loom" to find different looms)
3. Compare results by their values in the columns; you can also filter for specific licensing schemes (default: Look for a project with permissive licences)
4. Select your favourite result and click on its name to enter its project detail page (Default: OHLOOM)
5. Review the information on the project detail page (specifically the functional description on the top of the page)
6. click on "download bundle" to download files available for this version of the OSH project; click on "To XXX repo" to view the project repository for further information

The heuristic evaluation revealed problems in the areas of help and documentation, visibility of system status, user control and freedom, and consistency and standards.

In our view, the following three major problems need further attention:

1. The filtering for exploring data gets users stuck in seemingly unusable states.
2. Parts of the content still contain placeholder text.
3. Some bundle downloads do not work or demand a login.

#### 6.4.1.1 Filter Options

The filters to explore projects get the interface stuck in a state that makes them seem unusable from the user's perspective.

The selected values in the “Licence” and “Repo Host” dropdown are no longer visible after the

<sup>52</sup> Nielsen, Jakob. “Characteristics of Usability Problems Found by Heuristic Evaluation.” NN/g Nielsen Norman Group, 1 Jan. 1995, <https://www.nngroup.com/articles/usability-problems-found-by-heuristic-evaluation/>.

<sup>53</sup> Jeffries, Robin, et.a. “User Interface Evaluation in the Real World: A Comparison of Four Techniques” CHI '91: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Apr. 1991, pp. 119–124, <https://doi.org/10.1145/108844.108862>.

<sup>54</sup> Nielsen, Jakob. “10 Usability Heuristics for User Interface Design.” NN/g Nielsen Norman Group, 24 Apr. 1994, upd. 15 Nov. 2020, <https://www.nngroup.com/articles/ten-usability-heuristics/>.

dropdown is closed. This gives the impression that items cannot be found, although they are just filtered away.

We recommend adding a label outside of the dropdown.

When a value is selected it should be in the dropdown.

Consider a “reset” button that can be used instead of reloading the page when users get stuck.

In its default state, the organisation search box looks disabled due to the lighter grey hue. We recommend adding the darker “search” icon like in the search field or use a darker border.

When filtering by organisations, the value can be selected by typing and then pressing enter, but cannot be deleted by deleting the text. Instead the “x” button on the right needs to be pressed.

In the organisation filter, selection by scrolling and clicking is allowed. However, this amounts to manually scanning for the label, as items are not ordered in an obvious way:

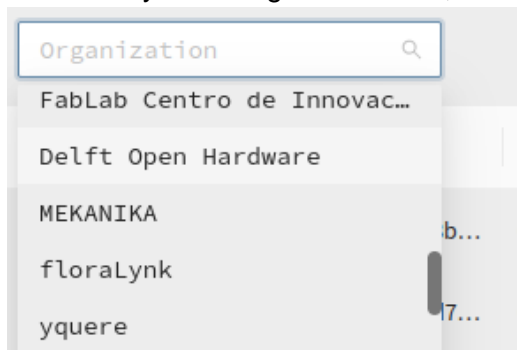


Figure 14: “Organisation” filter allowing selection by scrolling and clicking, but designed with a lighter grey hue making it appear disabled

We recommend ordering the items in the menu alphabetically for a better overview.

#### 6.4.1.2 Unclear System Status

The loading state is not clear. When the list of results is loading, “no data” is shown as if no data is available at all. This can lead to the assumption that there are no matching projects while, in fact, the results are not delivered yet.

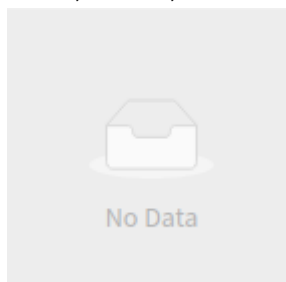


Figure 15: While loading a “no data” image is shown

We recommend showing a loading indicator.

Many items show a long, seemingly random letter/number text instead of a version number. This is the Git commit hash.

ol Laser Mini	d8ad0b9e8daa01582037b...	CC-BY
ol Laser Mini	d8ad0b9e8daa01582037b...	CC-BY

Figure 16: Items do not display version number, but Git commit hash

We recommend leaving it free or showing a placeholder.

Some items show empty name fields.

maquette #2 edition	758050125
	49a858125
	49a858125
08-TravailDécent	ee8a3e3ee

Figure 17: Items with empty name fields

This makes it difficult to know what the item to click on is actually about or relevant to the search. We recommend not permit nameless items or, at least, show a placeholder.

#### 6.4.1.3 Match between System and Real World

Repositories can be in any language, while the User Interface will usually be in English. For people who do not know this, the language mix is surprising. It could help to allow multilingual descriptions.

Whenever “repository” is meant on the website, the full word should be used instead of the abbreviation “repo”. While this is a common expert jargon, it might be unfamiliar to many users who also should be attracted to use the database.

#### 6.4.2 Usability Testing by Userlutions

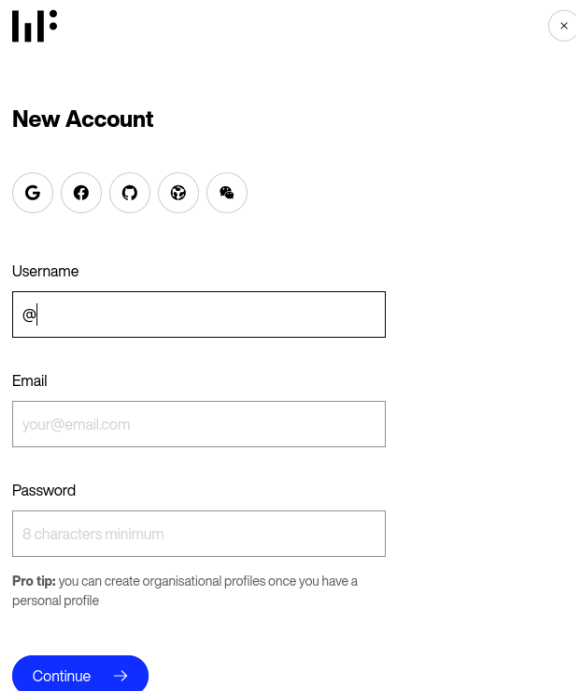
Userlutions conducted a usability check along the user journey for Wikifactory. As the method, a remote usability test was chosen with 50 users, equally split in men and women, interested in open source hardware and CAD.

Initially, the intention was to have more partners from the OPEN!NEXT ecosystem participating in the tests. For this testing purpose, a hiring meeting took place in Amsterdam in November for the partners. WMDE and FHG asked partners to participate in the testing activities. A special GDPR compliant survey was created and we collected ten answers to the request. In the end, when Userlutions was hiring skilled people for the testing, only three persons could participate and all other people were hired by the agency.

In the tests, four aspects were investigated: First impressions, user guidance, features, and profile.

## 6.4.2.1 First impression

The Wikifactory website is well structured and has a modern design, which helps users understand the purpose of the site. The [registration](#)<sup>55</sup> process works without problems.



**New Account**

Username

@

Email

your@email.com

Password

8 characters minimum

**Pro tip:** you can create organisational profiles once you have a personal profile

Continue →

Figure 18: Registration on Wikifactory.com

On the other hand, contact information is overlooked. For example, the icon for the chat option is very small.

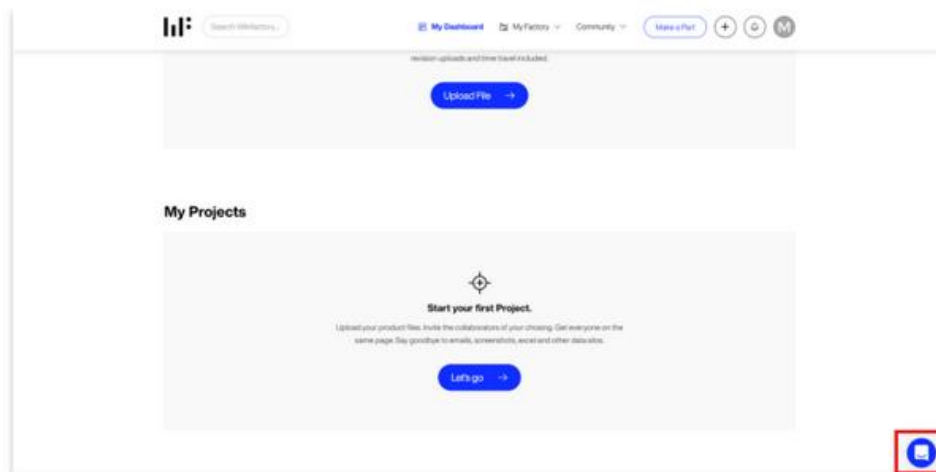


Figure 19: View of the website with the small icon for the chat option in the bottom right corner

And because of missing contact information, it is unclear for users how to contact customer service.

They recommend creating a separate contact-page with more contact options.

<sup>55</sup> <https://wikifactory.com/sign-up>

#### 6.4.2.2 User Guidance

The “+”-Icon doesn’t suggest that there is a menu behind it. That’s why the users find it difficult to get an overview of the features and don’t understand which features are offered. In addition, they are missing an onboarding which helps them to understand the website.

They suggest redesigning the menu button and replacing the icon with a more familiar menu icon. And to communicate more with the users by creating an onboarding for them and explaining what they can do on the website.

#### 6.4.2.3 Features

The users have trouble with the wording of different features and missing clear instructions, because of an inaccurate description. For example, the “Import a project” function isn’t even displayed on the dashboard, that’s why they can’t enable the function on the dashboard right away. Furthermore, the existing CTA “Lets go” only redirects them to create a project and not to import one.

Userlutions recommends placing functions visibly on the dashboard and renaming CTAs, by using a more specific wording and giving more explanations to the users about the features.

#### 6.4.2.4 Profile

Users have trouble adding skills to their profile, because the option is too small and not in focus of the users’ view.

Userlutions recommends placing the option “Add skill” closer to the profile picture & description to put it more into the focus of the users.

### 6.5 Summary

The initial grant agreement didn't consider a LOSH-Frontend. That substantial oversight was finally corrected in T3.4 with the finalisation of the frontend application. Still the conception of the goal in regards to the deliverable for the frontend has room for improvement:

In future setups, we would recommend not intertwining different goals too tightly for two reasons: First, from a research perspective, the approach of testing and populating the database at the same time distorts the insights for the evaluation considerably. Second, decoupling the processes allows for changing circumstances more easily. This also applies to the metrics that were defined upfront in the work package. Instead of setting fixed values (e.g. 500 participants in a hackathon or 50 people testing a software), it would have been more helpful to define the number of testings and connected iterations we would like to do in order to build useful and usable software.

Another remark that should be in mind when interpreting the evaluation is to acknowledge that software builds usually run through several testing cycles. The amount of issues found with this very comprehensive testing cycle is neither an indicator of a bad concept nor poorly built software but simply a normal result of a first round of testing.

The frontend is an integral addition to the LOSH, in order to facilitate the discovery and retrieval of OSH data from the ON Wikibase instance. It gives more people access and can act as a low-barrier entry point to the OSH communities.

In its current status, the frontend provides the foundation for future development. The testing gave valuable insights on what those future improvements in the workflows could look like to make it easier for future users to interact and engage with the built products. While the workflows, like entering skills on Wikifactory, were imagined to be useful for them, even people with close connections to the open source hardware community were struggling in this regard. On the other hand, quick improvements can be made, for example, just by changing the wording in “call to action”-buttons or by removing placeholder text for a clearer overview.

## 7 Summary and Outlook

The T3.4 goal is a populated database as an enhanced demonstrator. Two main channels

were used for the population: WikibaseReconcileEdit - a reconciliation API and the Krawler, a web crawler that automatically fetches data from repositories in regular intervals.

Both channels for populating the database are designed to work closely with each other. The Reconciliation API functions as the main conductor of data from sources such as the Krawler into the LOSH Wikibase instance. Due to its quality check, the WikibaseReconcileEdit automatically gives a minimum of quality assurance of the data added to the knowledge base by the Krawler.

The Krawler that enables data collection, processing and publication inside LOSH is a modular software solution. It bridges platforms that seem not to be linked to date. It works in conjunction with other submodules of which the following four were finalised in T3.4: The Appropedia scraper, the Repository Checker, the Manifest File Creator, and the LOSH Reporter. The submodules, too, are integral parts for populating the knowledge base with data increasing in quality and quantity.

The Manifest File Creator allows users to make OSH metadata available by helping to create or change manifest files for platforms that do not offer an API. The Appropedia Scraper that enhances metadata is a second means to increase the amount of OSH data available for pick-up by the Krawler. A quality check of corresponding OSH projects is done by the OSH Repository Checker that can identify deficient data and return a helpful report indicating areas for improvement to developers. Finally, the LOSH Reporter extracts statistical data on OSH projects and thus fills the gap left by OSHdata who no longer publish their annual statistical report on actors in the OSH community.

Populating the knowledge base only covers part of what is necessary to make the knowledge base a useful instrument for the OSH community. A low barrier option to retrieve and discover the available OSH data in the wikibase instance is needed in addition to the SPARQL endpoint that allows for complex queries.

The LOSH-Frontend provides this interface for the knowledge base. In its current state, the frontend provides a solid basis for further extension by the OSH community and for giving the affiliated communities even better access to available data on OSH products and services. With an attractive and easy to use entry point, we expect wider usage within the community and a steady growth of the ecosystem and the data relevant for them. Especially SMEs without specific knowledge in SPARQL can now easily browse available OSH solutions and build new business cases.

The populated knowledge base functions as a hub for OSH metadata and links open source and commercial communities and their products. Its aim is to be accepted by actors in the OSH ecosystem whether they are business or open source community members. The Appropedia Scraper is an example of a best case collaboration scenario between LOSH and OSH online platforms, which feature the primary data source for the knowledge base.

To facilitate the sustainability of the ON knowledge base beyond the ON project three aspects were considered:

- Adaptable modules
- Low maintenance
- Available documentation

The developed modules, the reconciliation API and the frontend application are all designed in such a way that they can easily be adopted to various use cases. Documentation is available to interested parties. Specifically the Krawler requires only minimal maintenance to facilitate



future hosting by the communities. The ON knowledge base offers a starting point to improve OSH data quality and move towards a common data structure.

In the upcoming T3.5 - the demonstrators of WP3 will be validated. The validation will happen in collaboration with the SME and OSH community who will interact with the platform in an event like a hackathon. The plan is that there participants will be learning how to use LOSH and Wikifactory developed tools, how to use and where to find documentation, and how to upload and extract data that is necessary for participants.

We expect the participants to form a community around the platform and integrate it into their OSH activities.

## Bibliography

Bonvoisin, Jérémy, et al. “Standardisation of Practices in Open Source Hardware.” *Journal of Open Hardware*, vol. 4, no. 1, 15 Apr. 2020, <https://doi.org/10.5334/joh.22>.

Bonvoisin, Jérémy, et al. “Vielfalt Und Stand Der Open-Source-Hardware.” *Interdisziplinäre Perspektiven Zur Zukunft Der Wertschöpfung*, 2017, pp. 121–133., [https://doi.org/10.1007/978-3-658-20265-1\\_10](https://doi.org/10.1007/978-3-658-20265-1_10).

Ezaji, A., et al. “Requirements for Design Reuse in Open-Source Hardware: A State of the Art.” *Procedia CIRP*, vol. 100, 2021, pp. 792–797., <https://doi.org/10.1016/j.procir.2021.05.042>.

Jeffries, Robin, et.a. “User Interface Evaluation in the Real World: A Comparison of Four Techniques.” CHI '91: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Apr. 1991, pp. 119–124, <https://doi.org/10.1145/108844.108862>.

Matheus, Ricardo, et al. “Open Government Data and the Data Usage for Improvement of Public Services in the Rio De Janeiro City.” *Proceedings of the 8th International Conference on Theory and Practice of Electronic Governance*, 2014, <https://doi.org/10.1145/2691195.2691240>.

Mies, R., et al. “Development Of Open Source Hardware In Online Communities: Investigating Requirements For Groupware.” *Proceedings of the Design Society: DESIGN Conference*, vol. 1, 2020, pp. 997–1006., <https://doi.10.1017/dsd.2020.38>

Moritz, Manuel, et al. “Best Practices and Pitfalls in Open Source Hardware.” *Proceedings of the International Conference on Information Technology & Systems (ICITS 2018)*, 2018, pp. 200–210., [https://doi.org/10.1007/978-3-319-73450-7\\_20](https://doi.org/10.1007/978-3-319-73450-7_20).

Nielsen, Jakob. “10 Usability Heuristics for User Interface Design.” NN/g Nielsen Norman Group, 24 Apr. 1994, upd. 15 Nov. 2020, <https://www.nngroup.com/articles/ten-usability-heuristics/>.  
— “Characteristics of Usability Problems Found by Heuristic Evaluation.” NN/g Nielsen Norman Group, 1 Jan. 1995, <https://www.nngroup.com/articles/usability-problems-found-by-heuristic-evaluation/>.

“Din Spec 3105-1:2020-07.” *DIN SPEC 3105-1 - 2020-07 - Beuth.de*, <https://www.beuth.de/de/technische-regel/din-spec-3105-1/324805763>.

## 8 Annex

### 8.1 Further Information on the Krawler

#### 8.1.1 Data Sources of the Krawler

##### 8.1.1.1 Requirements

LOSH aims to cover a large part of the OSH designs already published online. The identified target group of users will likely expect a certain variety of OSH designs, a reasonable quality of those designs and meaningful metadata about found OSH designs. Aside from technical prerequisites, these have been the three main perspectives taken into consideration for the selection of data sources for the Krawler. 22 OSH online platforms have been analysed, please find the spreadsheet with the details<sup>56</sup> of this analysis attached to this report (see “Platform\_List\_OKH\_LOSH.csv” in the subfolder “raw data”). The following paragraphs give a summary of the conclusions made for each perspective.

**Quality-wise**, OSH designs are published in very distinct qualities, sometimes with incomplete documentation<sup>57</sup> or under licensing terms that prohibit commercial exploitation or the publication of derivatives<sup>58</sup> (which is, by definition, not open source<sup>59, 60</sup>). This has been reflected in the research carried out in the ON project<sup>61</sup> and previous research projects like ON.<sup>62</sup>

Effective reuse of a given OSH design requires its complete technical documentation, while the concretely necessary information depends on several factors, such as the technology embedded in the design, the envisioned use case of the documentation and its target group (DIN SPEC 3105-1), (Ezoi et al.). However, the OSHWA certification program and community-based assessment according to DIN SPEC 3105-2 define those requirements at a reasonable level that makes OSH designs comparable.<sup>63</sup> DIN SPEC 3105-1 deems a documentation release complete when specialists in the corresponding field of technology can understand, modify and replicate the OSH design only by the given technical documentation.

<sup>56</sup> [https://github.com/OPEN-NEXT/LOSH-list/blob/main/Platform\\_List\\_OKH\\_LOSH.csv](https://github.com/OPEN-NEXT/LOSH-list/blob/main/Platform_List_OKH_LOSH.csv)

<sup>57</sup> Bonvoisin, Jérémy, et al. “Vielfalt Und Stand Der Open-Source-Hardware.” *Interdisziplinäre Perspektiven Zur Zukunft Der Wertschöpfung*, 2017, pp. 121–133., [https://doi.org/10.1007/978-3-658-20265-1\\_10](https://doi.org/10.1007/978-3-658-20265-1_10).

<sup>58</sup> Moritz, Manuel, et al. “Best Practices and Pitfalls in Open Source Hardware.” *Proceedings of the International Conference on Information Technology & Systems (ICITS 2018)*, 2018, pp. 200–210., [https://doi.org/10.1007/978-3-319-73450-7\\_20](https://doi.org/10.1007/978-3-319-73450-7_20).

<sup>59</sup> <https://certification.oshwa.org/process/hardware.html>

<sup>60</sup> “Din Spec 3105-1:2020-07.” *DIN SPEC 3105-1 - 2020-07 - Beuth.de*, <https://www.beuth.de/de/technische-regel/din-spec-3105-1/324805763>.

<sup>61</sup> Ezoi, A., et al. “Requirements for Design Reuse in Open-Source Hardware: A State of the Art.” *Procedia CIRP*, vol. 100, 2021, pp. 792–797., <https://doi.org/10.1016/j.procir.2021.05.042>.

<sup>62</sup> <https://opensourcedesign.cc/>

<sup>63</sup> Bonvoisin, Jérémy, et al. “Standardisation of Practices in Open Source Hardware.” *Journal of Open Hardware*, vol. 4, no. 1, 15 Apr. 2020, <https://doi.org/10.5334/joh.22>.

**Diversity-wise** however, also OSH designs with incomplete technical documentation can be useful, specifically in the ideation phase of product development. Essential parts may be copied, reworked or completed or simply serve as inspiration for an own approach. As LOSH aims to offer a general purpose meta search for OSH designs, users will likely expect a knowledge base of OSH designs covering very diverse technology fields. To be included in LOSH's database, an OSH design must at least comply with the following terms:

- published under a free/open licence, alongside with
- a readme and
- at least one source file (so a file different from standard files like CONTRIBUTING or CODE-OF-CONDUCT and images)

The **data quality** of extracted metadata significantly differs among data sources. While automatic input via platform APIs may be the most convenient for users (since no manual interaction with the LOSH system is needed for this step) data quality fully depends on such platform APIs. Naturally, full compliance with the OKH-LOSH specification is, by the current state, only possible by the use of manifest files. A focus on data quality is needed to provide users with the full information available about OSH designs and to test the OKH-LOSH specification in practice.

### 8.1.1.2 Results

Table 1 shows all current data sources for LOSH and the perspectives (OSH design quality and diversity and data quality) covered by that source.

Data Source	Metadata Transfer Method	OSH design quality	OSH design diversity	Data quality	URL
GitHub	Manifest files		x	x	<a href="https://github.com/">https://github.com/</a>
OSHA Certification List	API	x			<a href="https://certification.osha.org/list.html">https://certification.osha.org/list.html</a>
Wikifactory	API		x		<a href="https://wikifactory.com/">https://wikifactory.com/</a>
Thingiverse	API		x		<a href="https://www.thingiverse.com/">https://www.thingiverse.com/</a>
Appropedia	Manifest files, collected by a scraper, uploaded to GitHub		x	x	<a href="https://appropedia.org/">https://appropedia.org/</a>
OKH-Search	Manifest files, collected by a scraper, uploaded to GitHub		x	x	<a href="https://search.openknowhow.org/">https://search.openknowhow.org/</a>
manually curated data set <sup>64</sup>	Manifest files in a dedicated repository on	x		x	<a href="https://github.com/OPEN-">https://github.com/OPEN-</a>

<sup>64</sup> containing OSH designs from the design reuse guideline and direct recommendations from the OPEN!NEXT consortium and the OSH community

	GitHub				NEXT/LOSH-list/tree/main/manifest_files/manually-created
OSEG-CAB	Manifest files in a dedicated repository on GitHub	x		x	<a href="https://github.com/OPEN-NEXT/LOSH-list/tree/main/manifest_files/manually-created/OSEG-CAB">https://github.com/OPEN-NEXT/LOSH-list/tree/main/manifest_files/manually-created/OSEG-CAB</a>
OHO-CAB	Manifest files in a dedicated repository on GitHub	x		x	<a href="https://github.com/OPEN-NEXT/LOSH-list/tree/main/manifest_files/manually-created/OHO-CAB">https://github.com/OPEN-NEXT/LOSH-list/tree/main/manifest_files/manually-created/OHO-CAB</a>

*Table 1: Overview and reference to current data sources for the LOSH-Krawler alongside with the perspectives they mainly cover*

High-quality OSH designs are mainly found in dedicated certification programs, specifically the OSHA certification program and conformity assessment bodies (CAB) conduction community-based assessment of OSH designs according to DIN SPEC 3105-2, with Open Source Ecology Germany e.V. (OSEGeV) and Observatory of Open Hardware e.V. (OHOeV) being the two active CABs. Apart from that, other, not (yet) certified, but nevertheless high-quality OSH designs have been included in a manually curated data set. Proposals for this data set have been identified during the research carried out within the ON project for the guideline for design reuse in T2.3 “Facilitating Documentation for design reuse”.<sup>65</sup> Furthermore, the ON consortium and members from the OSH community gave recommendations for high-quality OSH designs.

To foster diversity in LOSH's data set, while assuring a reasonable quality and informative value of data, automatic input streams from the Wikifactory platform, Appropedia, Thingiverse and projects following the OKHv1 specifications have been established. Some connections could be made directly using the platform API (as in the case of Wikifactory or Thingiverse), others relied on the use of additional scripts. While both ways deliver satisfying data in an automated process, the latter is more prone to disruptions and hence potentially requires more maintenance in the long run.

Wikifactory as a dedicated platform for collaborative OSH development offers a wide range of different technologies, but only 11.7 % of OSH designs would be freely (including commercially) exploitable under a free/open licence. This has been shown in a test crawl<sup>66</sup> (the

<sup>65</sup> <https://github.com/OPEN-NEXT/WP2.3-Guideline-and-template-for-documentation-of-OSH-design-reuse>

<sup>66</sup> The crawl performed on the Wikifactory API on May 6th 2021 returned 3990 projects for which 3523 had no mention of a licence, 2554 had no image and 36 had a licence not matching with any free/open licence on the SPDX licence list.

data can be accessed in the appended zip archive “WIF Test Crawl” in the subfolder “raw data”).

Thingiverse was selected for the sheer quantity of OSH designs from different technology domains (mainly 3D printing and electronics), often designed to be replicated in a DIY context. By April 6th Thingiverse was hosting 5.343.156 different designs. A random sample was drawn and showed that about every fifth Thingiverse project would qualify for an upload to LOSH.<sup>67</sup> Appropedia, one of the developers and early adopters of the OKHv1 specification<sup>68</sup> offers freely exploitable OSH designs only, however the platform struggles with a large quantity of unsorted content in the domain of appropriate technologies, reaching from OSH designs, over how-to guides for the usage of necessary tools to educational material about permaculture in critical environments. Relevant OSH designs are identified and made available through manifest files in a semi-manual process by AI-supported data analyses of their database and the Appropedia community.

Moreover, OSH designs following the OKHv1 specification (apart from Appropedia projects) were crawled. On the one hand, this step completed the downward compatibility to OKHv1, which is essential to keep the OKH-LOSH specification and the LOSH project relevant for the OSH community. On the other hand, this step was partly realised by making git-based platforms (specifically GitHub and GitLab) searchable for the crawler. In a study, git-based platforms have been identified to be currently the best option for open source development processes (Mies et al.), hence LOSH supports this go-to-reference.

## 8.2 Setup and Usage of the Krawler

### 8.2.1 Requirements

- Python v3.6 or later
- Poetry v1.1 or later (for packaging and dependency management)
  - find installation instructions for poetry here: <https://python-poetry.org/docs/>

### 8.2.2 Installation Instructions

The modules of the Krawler are written in Python and are all executable on a bash level – either locally on the user’s machine or on a server e.g. using cron (a job scheduler on Unix-like operating systems). Consequently downloading the files from the public GitHub repository of the Krawler<sup>69</sup> and configuring the Krawler as explained below will suffice. Besides that, a video is provided outlining design notes, usage and installation instructions<sup>70</sup>.

#### Poetry

Once you have `poetry` in your `PATH`, install the project by entering the `krawl` directory (where the `pyproject.toml` file is located) and type:

<sup>67</sup> Most projects use non-free licences (such as CC BY-NC or CC BY-ND prohibiting commercial use or the creation of derivatives). About 20% of repositories are mostly empty or not accessible anymore.

<sup>68</sup> Open Know-How Specification. (2022). *Internet of Production Alliance*. Retrieved from <https://standards.internetofproduction.org/pub/okh>

<sup>69</sup> <https://github.com/OPEN-NEXT/LOSH-krawler> (GPL-3.0-or-later)

<sup>70</sup> <https://ipscloud.ipk.fraunhofer.de/index.php/s/s4L7i3EE39gmjGR> (CC-BY-4.0, Alec Hanefeld)

```
sh
poetry install
poetry shell
```

All commands in the `bin/` directory expect the `poetry shell` to be active.

### Credentials

All environment variables in the `.env.example` file must be defined<sup>71</sup>, see the code block below:

```
# This can be any directory where the krawler can write its
intermediary files,
# (just make sure it is not in a git dir or ignored by git,
# as it will contain a lot of files):
KRAWLER_WORKDIR="..."

# Wikibase OAuth client
KRAWLER_WB_CONSUMER_KEY="..."
KRAWLER_WB_CONSUMER_SECRET="..."
KRAWLER_WB_ACCESS_TOKEN="..."
KRAWLER_WB_ACCESS_SECRET="..."
# (... ask your wikibase admin if you are unsure how to get
this)

# Wikibase Reconciler Prop ID
# This is the id (ie. P123) of a wikibase property of type URL
and ideally name "id".
# This property has to be created in wikibase and then set
here.
KRAWLER_WB_RECONCILEPROPID="..."

# GitHub specific env variables
KRAWLER_GITHUB_KEY="..."
# (... get one [here] (https://github.com/settings/tokens))
```

## 8.2.3 Execution

Every module and also every specific fetcher can be executed individually calling the corresponding scripts (see below). Likewise data collection jobs and post-processing can be easily customised per job.

### 8.2.3.1 Data collection (Fetcher)

To fetch project from GitHub.com

```
sh
krawl/bin/fetch/github/gh.sh
```

---

<sup>71</sup> You might want to use <https://direnv.net/> to automatically source the `.env` file you created.

To fetch project from GitLab.com

To fetch project from Gitlab.OpenSourceEcology.de

To fetch projects from Wikifactory.com

```
sh
krawl/bin/fetch/wikifactory/wf.sh
```

To fetch project from the OSHWA Certification List

To fetch project from Thingiverse.com

### 8.2.3.2 Convert data into Linked Open Data (RDF-Converter)

To convert all TOML to RDF (turtle, ttl)

```
sh
krawl/bin/push/wikibase.sh
krawl/bin/push/rdf.sh
```

### 8.2.3.3 Data upload (Wikibase-Pusher)

To push all found projects to wikibase<sup>72</sup>:

```
sh
python -m krawl.wikibase.core
```

If you want to push an individual `ttl` file you can also use:

```
sh
python -m krawl.wikibase.core ./samples/okh-sample-
OHLOOM_fixed.ttl
```

## 8.2 Additional Findings from the Heuristic Evaluation

*PDF file with the name “2022-3\_LOSH Heuristic Analysis Results.pdf” with findings is uploaded to the EUC website together with this report as a separate file.*

## 8.3 Full Report on Usability Testing by Userlutions

*PDF file with the name “UX-Report\_Userlutions\_Wikimedia.pdf” with analysis and findings and PDF file with the name “Recommendation-Checklist\_Userlutions\_Wikimedia.pdf” with recommendations are uploaded to the EUC website together with this report as separate files.*

---

<sup>72</sup> Note that the pusher currently always pushes every file it finds, even if it has already been pushed. Duplicates are filtered by the WikibaseReconcileEdit module.