LUND UNIVERSITY
Faculty of Science

Making conference presentations searchable and citeable

Christian Cederholm

Thesis submitted for the degree of Bachelor of Science

Project duration: 6 months, half time
Examination: Spring 2020

Supervised by Caterina Doglioni

Department of Physics
Division of Particle Physics
June, 2019

Abstract

Indico is the main conference event manager where talks in high energy physics are published online. Indico allows to access presentation from conferences, but it does not yet organize them coherently or make them easily citable by others. To solve this, we want to write code for Indico users that saves not only presentations used during conferences but also their metadata. This will make it easier to find the presentations, going from looking for a needle in a haystack to searching for a book in the library. The code from this thesis will upload the presentation and its metadata to its own web page on the Zenodo repository, which will give it its own personal Digital Object Identifier (DOI), which is the equivalent of a social security number for a digital document. The Zenodo webpages created will show how many times the work has been viewed or downloaded, it tells the reader which version is uploaded in case of multiple versions, contains the date it was uploaded, shows information and links about the conference where the work has been presented and which groups it might interest, presents information about the license it is shared under and is indexed in the European Open Science Infrastructure, for open scholarly and scientific communication (OpenAIRE) so people can search for it.

Contents

List of Abbreviations

DOI = Digital Object Identifier
FAIR = Findable, Accessible, Interoperable and Reusable
HEP = High Energy Physics

Chapter 1

Introduction

The aim of this project is to make a computer program which will make it possible for the part of the research community that uses a website where talks are uploaded [1] to upload not only the presentations which are also called talks used during conferences but also the metadata of the presentations. Both presentations and their metadata will be uploaded to a website [2] and assigned a standardised unique identifier [3].

Why is this work necessary: In for instance high energy physics conferences, neither conference management or information systems assign conference presentations a unique identifier. This makes it hard to cite the talks after the conference. Moreover, the authors of the presentations also cannot see how many times their work has been utilised or cited. Seeing how many times a work has been cited is important as it is a significant way to show the works relevance if the author of the work e.g. is applying for grants or employment. It is also inconvenient for each author from a given conference to upload their talk to a data saving platform individually. Especially if they have already uploaded their talk through a conference management system such as Indico [4].

Solution in this project: The solution used in this project is to archive these talks to a website which will assign them a unique identifier and a web-page. The web-page:

- Contains information on how many times the talk has been viewed or downloaded.
- Tells the reader which version of the talk is uploaded in case of multiple versions.
- Contains the publication date of the talk.
- Contains information and links about the conference where the talk has been presented and which communities it is relevant to.
- Contains information about the license under which the talk is distributed.
- Is indexed [5] so people can find the talk using a search.

This project will provide scripts that allow conference presentations and their metadata to be exported from the Indico conference management system and imported into Zenodo. The code was built using the Zenodo Rest API[6] as a basis.

Chapter 2

Background for this project

Openness and accessibility are relevant to the work in this project. The material from the conferences that I have written code for in this work is open to anyone, meaning that it is freely and publicly available [7]. However, not all the material and its accessory information (metadata) is easily accessible yet.

In this chapter, I will summarize the FAIR principles as they are crucial principles for any scholarly data and derived products that are meant to be publicly available and easily found. I will then describe the Indico event manager that is used to host the material presented in high energy physics conferences, and describe the Zenodo repository where the code developed in this project will host the material and the metadata to make it easily accessible and citeable. Finally, the presentation information and its metadata that my code will import from Indico to Zenodo will be described.

2.1 The FAIR principles

In FAIR [8], F stands for findable, A for accessible, I for interoperable and R for reusable [8]. To be *findable* means being easy to find when searched for. An object being *accessible* means that it is easy to access. An example of being easy to find but hard to access would be non-open access publications since they are easy to find but require payment to read. *Interoperability* is the ability of data or tools to work together. For example trying to view a pdf file in a text editor would only work if the pdf file did not have any pictures since the text editor does not have the capacity to show images. Finally *reusability* is how easy it is to reuse. If there is data of what percentage of animals in a number of countries has fur, and this data is used somewhere else to analyse how average temperature affects how many animals have fur, then that is an example of data being reused. Also, an article that is cited is an example of said article being reused. This means it is important that any scientific effort that aims to be open and available adheres to the FAIR principles.

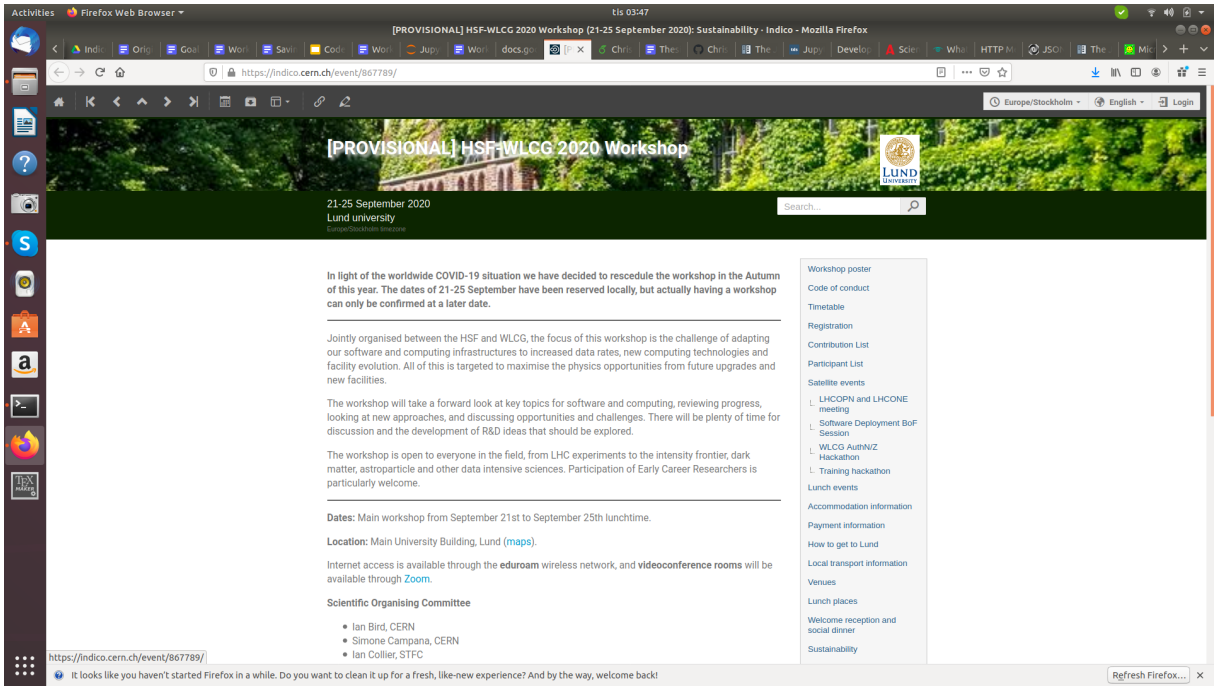
2.2 The Indico event manager

Indico [4] is an online event manager, meaning that it is used to host and manage information about conferences and lectures. Indico holds the information on the conference but also the conference material itself, such as pdf files created by the conference attendants

or videos recorded during the conference. An entry in the conference material is called a *conference contribution*. These conference contributions are saved on Indico, but their metadata is not easy to search for and is not citable.

An example of conference hosted by Indico is the High Energy Physics Software Foundation and Worldwide Computing Grid workshop[9] shown in Fig. 2.1, which should have been held in May 2020 in Lund but was then postponed due to COVID-19. In this project, I will use a sample contribution from this conference for prototyping the code, as shown in Fig. 2.2.

Figure 2.1: The front page of the High Energy Physics Software Foundation and Worldwide Computing Grid workshop conference.



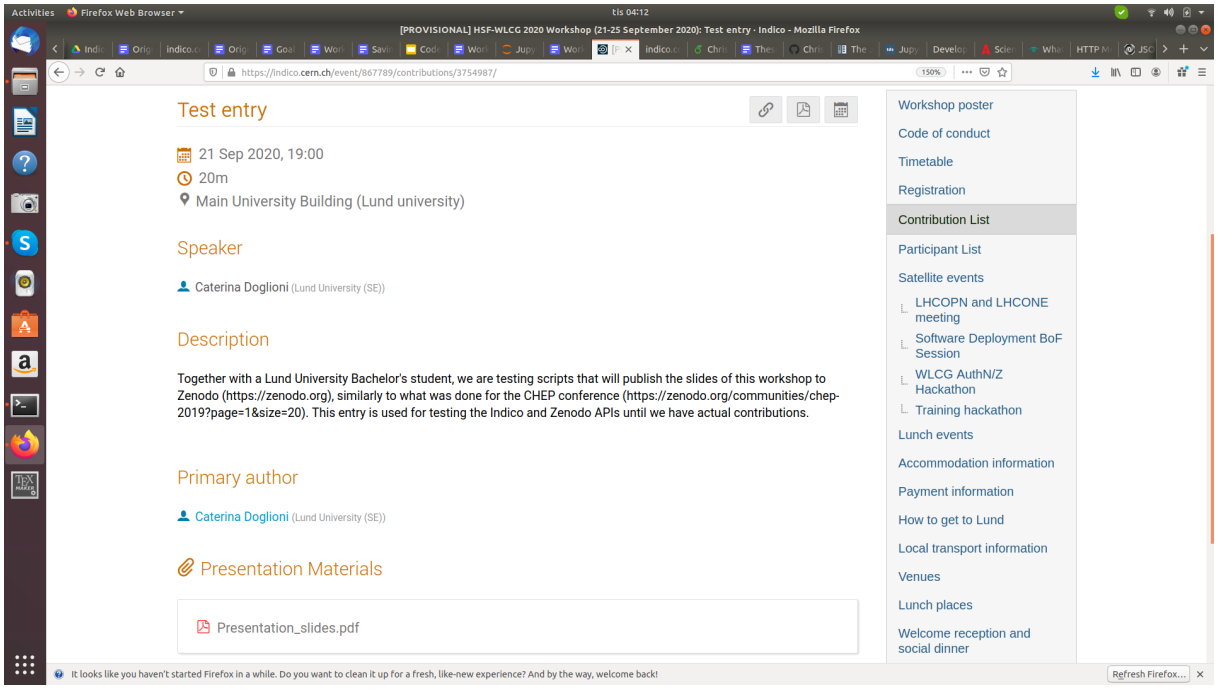
In Indico the metadata from presentations is not easily searchable or citable at the moment. The reason for this is that Indico only offers a link to the pdf and the authors name, rather than a unique identifier for the contribution.

2.3 The Zenodo platform

Zenodo [2] is an online repository for research publications, data and software which is free and publicly available. Zenodo gives any digital work uploaded to it its own web-site and unique personal identifier (Digital Object Identifier)[3], which makes it citable. An example of a Zenodo upload is shown in Fig. 2.3. It is also simple to search for works in Zenodo. The Zenodo entry contains all information filled in by the person who uploaded the work as well as the number of times the work has been viewed and downloaded.

There is also *sandbox.Zenodo* which is exactly the same as the regular Zenodo platform except for the fact that works can be edited after they have been published and given their own website. Shown in Fig. 2.4

Figure 2.2: The example contribution used when testing the code for this work. The metadata is the title at the top, the date just underneath it, the description, the author with the authors affiliation and the presentation file.



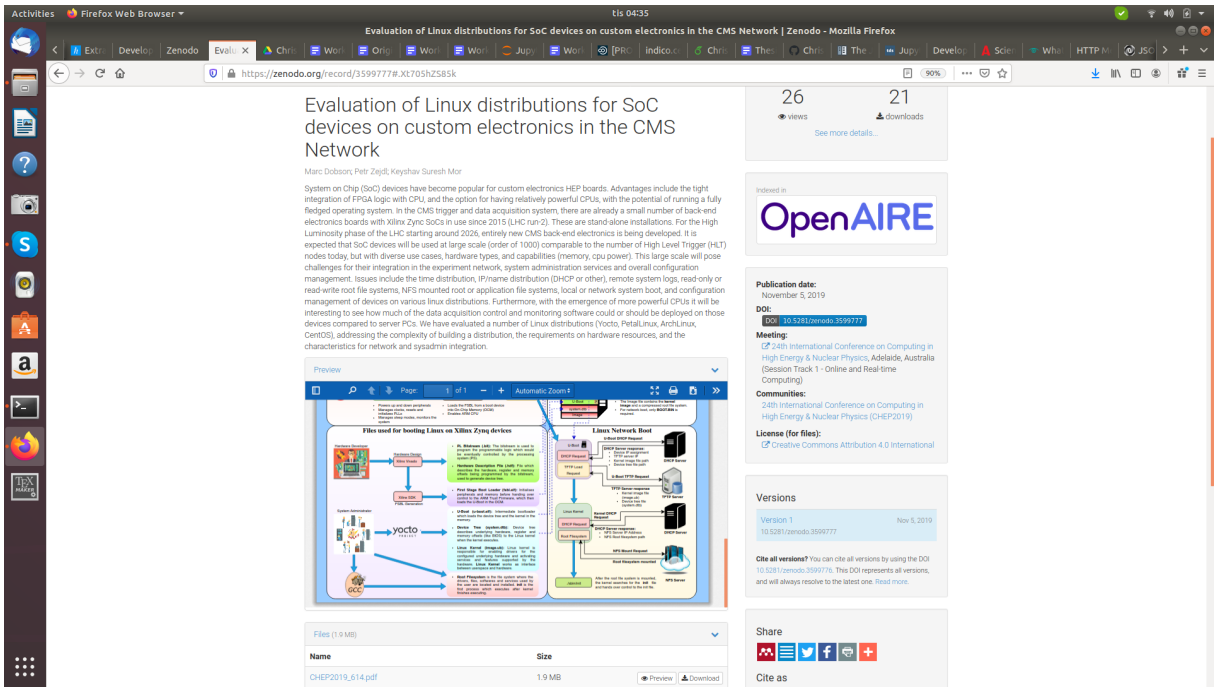
Zenodo is where the information on conference material from Indico will be uploaded.

2.4 Information needed to identify a conference contribution

The data that identifies a conference contribution, taken from Indico and posted on Zenodo includes the presentation title, date, author names and affiliations, description and the presentation itself.

- Title: The title of the work is the easiest way to find the work when searching for it. The title also indicates what the work is about so it should be included.
- Date: The date informs the reader how old the information in the presentation is. The date also aids when searching for a specific time period or if there are many similar works it helps to identify the order of the works.
- Author names: Who worked on the project is essential to include so that authors are given credit for their work. It will also facilitate searching for works by the same author.
- Author affiliations: The authors affiliations are included since knowing which institute the author/s are part of helps determine the credibility of the work. A work concerning particle physics whose author works at CERN is more credible than an

Figure 2.3: A Zenodo upload. The number of views and downloads can be seen in the top right corner and the DOI under publication date which is under OpenAIRE.



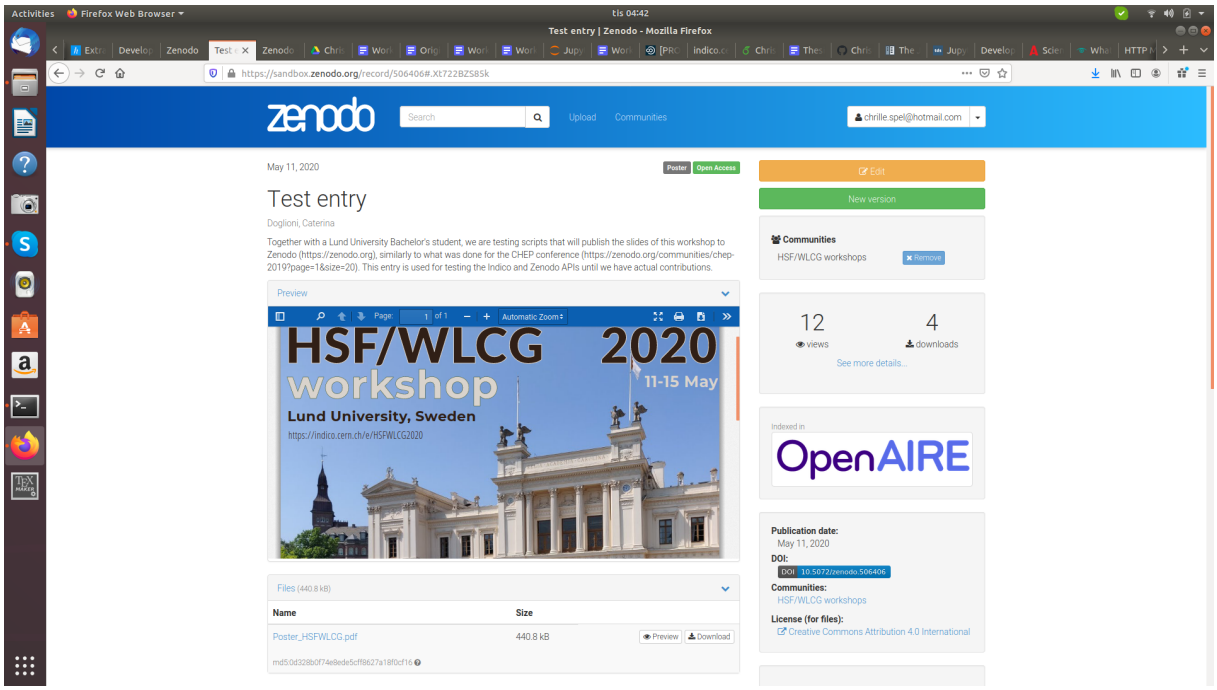
author who works at a bakery. Furthermore it can show how much research an institute does if one looks at the percentage of works where one or multiple author were part of said institute.

- Description: The description is helpful as it is more informative than the title and lets the reader know the focus of the work, which is helpful if the covered topic is broad or complicated. Details mentioned in the description but not in the title can be caught in searches.

- Presentation: Having the presentation be available is necessary since it is the most important document the entry refers to.

- Presentation title: The title of the work should be informative enough but if there are more files it helps to know which file is the presentation file.

Figure 2.4: An example of an upload in sandbox.Zenodo. The edit button does not exist in regular Zenodo since only `sandbox.Zenodo` allows editing of already uploaded works.



Chapter 3

Tools

This chapter will describe all the tools used for this project. Python will be explained as it is the programming language used for all the code written in this work. Python is also the programming language used by Jupyter notebook [10] which was used to make it easier to modify and explain the code, as well as making it more user friendly. I will also describe the JSON language [11] that Indico uses to export results about its conference. Lastly I will explain Zenodo sandbox along with its application to upload the presentation and its metadata called REST API. The Zenodo sandbox was used, as it is a version of Zenodo where entries can be edited even after being uploaded.

3.1 Python

Python is the high-level programming language that is used in this project to import information from Indico to Zenodo. The main reasons Python is the programming language used in this project is that Python is free and open source, as well as easy to read and learn [12]. Python also works on different platforms and is used for mathematics, working with databases [13], internet protocols [14] (sending data across the web) and other things.

3.2 Jupyter Notebook

In this project, I use Jupyter Notebook to show a prototype of my code in a user-friendly manner, using the Python programming language. [10]

Jupyter Notebook is a combination of two elements. The first element is a web application, which is a browser-based tool that allows its users to visualize text, formulas, computations and rich media. The second element is the Notebook document itself, which contains the code to be executed and shows the output of every component available in the web application, computations, explanatory text, images, etc. [15] Some of the main features of the web application are the possibility of editing code in the browser, executing code from the browser and seeing the results of any computations together with the code. Notebook documents contain the inputs and outputs of an interactive session, along with supplementary text which is not meant to be executed written together with the code. Notebooks can be downloaded in a number of different ways, such as HTML, LaTeX,

PDF, and slide shows.

Despite Jupyter being used in a web browser, Jupyter is actually running on the user's own computer. Should the web address begin with `http://localhost:` or `http://127.0.0.1:`, the user's computer is acting as the server. Jupyter can also be used remotely: a company or university might run the server for its employees/students. The basics of Jupyter notebook can be found in for example in Ref.[16].

The Jupyter notebook in this project is used to show how to manipulate data outputs from Indico and then place them in a format that can be uploaded to Zenodo. The information from Indico is extracted in the JSON language [11], and it is uploaded to Zenodo using the REST API [17]. JSON and the REST API are discussed in the following sections.

3.3 JSON

JSON stands for JavaScript Object Notation [11]. It is is a text file format used to export and import data, for example from Indico. It is easy for programs to read and create JSON files because of the structured design of those files. A JSON file consists of name/value pairs which are often called JSON objects. It also has an ordered list of values which in most programming languages is translated as a list, sequence or an array. A JSON object is of the form `("name":value)`. The `name` is a string and identifies the value, and the `value` can be string, number, boolean, null, array or another JSON object. This means a JSON object can be inside another JSON object. An example of JSON that Indico produces for a conference contribution entry is shown below. The example has a great deal of removed metadata, the removed metadata is not used in this project.

```
{
  'startDate':
    {'date': '2020-09-21',
    'tz': 'Europe/Zurich',
    'time': '19:00:00'},
  'endDate':
    {'date': '2020-09-21',
    'tz': 'Europe/Zurich',
    'time': '19:20:00'},

  'primaryauthors':
    {'person_id': 4830880,
     'affiliation': 'Lund University (SE)',
     '_type': 'ContributionParticipation',
     'last_name': 'Doglioni',
     'db_id': 4910219,
     'emailHash': '84e58ea01103c331c26e112c0f44c901',
     '_fossil': 'contributionParticipationMetadata',
      'fullName': 'Doglioni, Caterina',
     'first_name': 'Caterina',
     'id': '4910219'},

  'title': 'Test entry',

```

```
  'folders':
      {'_type': 'folder',
     'attachments':
        {'_type': 'attachment',
        'description': '',
        'content_type': 'application/pdf',
        'id': 3315426,
        'size': 440824,
        'modified_dt': '2020-02-26T15:39:11.211826+00:00',
        'title': 'Presentation_slides.pdf',
        'download_url': 'https://indico.cern.ch/event/867789/
           contributions/3754987/attachments/1989047/3315426/
           Poster_HSFWLCG.pdf',
        'filename': 'Poster_HSFWLCG.pdf',
        'is_protected': False,
        'type': 'file'},
     'title': None,
     'is_protected': False,
     'default_folder': True,
     'id': 1989047,
     'description': ''},

  'description': "Together with a Lund University Bachelor's student, we
      are testing scripts that will publish the slides of this workshop
     to Zenodo (https://zenodo.org), similarly to what was done for the
     CHEP conference (https://zenodo.org/communities/chep-2019?page=1&
     size=20). This entry is used for testing the Indico and Zenodo APIs
      until we have actual contributions.",

  url': 'https://indico.cern.ch/event/867789/contributions/3754987/'
}
```

3.4 Zenodo Sandbox and REST API

Zenodo accepts data to be uploaded from users using a REST API[17] [18]. The purpose of the REST API is to allow its users to receive and send resources (files, data, etc). REST API has four functions:

- GET: lets the user receive resources from an online storage space.
- POST: creates a new resource on an online storage space.
- PUT: updates a resource on an online storage space.
- DELETE: removes a resource on an online storage space.

The code used GETs data from Indico and then PUTs and POSTs it on Zenodo. To test the code in this project we used `sandbox.Zenodo` instead of the regular Zenodo website, as the `sandbox.Zenodo` uploads can be edited after being published. The code created and discussed in the next chapter uses a set of calls to the Zenodo REST API that are found on the Zenodo REST API tutorial [6].

Chapter 4

Work performed in this project

This chapter will explain the work performed during this project. The preliminary work consisted of coding tutorials, and was necessary to understand how to write the code. The code to upload the metadata from Indico to Zenodo will be described as it is the focus of this project. The issue of validating the code will be discussed as it is the greatest problem faced during this project. The last point will be implementing multi-entry support in the code. This is necessary as the code may very well fail if it is only able to process one entry at a time.

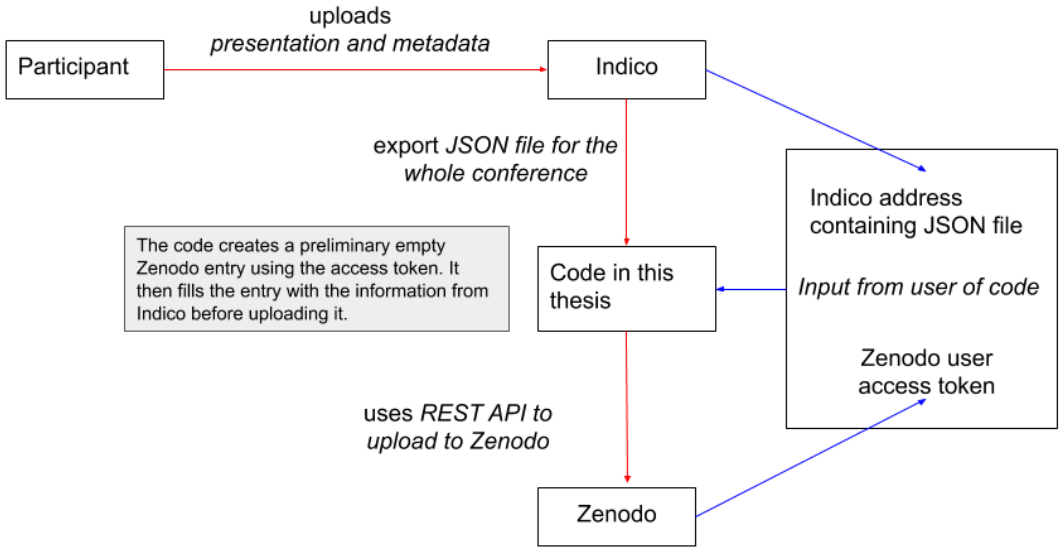
4.1 Preliminary work

The work that I performed prior to the creation of the code was mainly learning how to use a terminal, Python and Jupyter Notebook detailed in the previous chapter. The code was modified using the Zenodo REST API as a basis.

4.2 Code to upload Indico data to Zenodo

The code starts by finding and retrieving the metadata from Indico in the form of a JSON file. The Indico address where the JSON information is stored must be provided by the user. The user must also download the presentation .pdf locally. The metadata is retrieved by a call to the Zenodo REST API using `requests.get` and the conversion to JSON file is done using a `.json()` command found in the Zenodo REST API tutorial. Secondly, the code copies the desired metadata and stores them in memory. The code then searches for the name of the JSON objects we want in Zenodo and copies its value. Some metadata is inside a subgroup of a subgroup. For example if one looks at the example JSON file in Chap. 3.3 the value of the "startDate" name is 3 JSON objects, "date","tz" and "time". But in the example we are already inside a subgroup of a subgroup of a subgroup etc. Thus since the code must enter the correct subgroup of a subgroup of a subgroup etc to find the correct metadata, it requires all files to be of the same format. The code then creates a preliminary empty Zenodo entry. This is done with a `requests.post` command as shown in the REST API tutorial. Next, the code inserts all stored metadata into the empty Zenodo entry. Which is performed using a `requests.put` command also shown in the REST API tutorial. Finally, the code will publish the Zenodo entry. This is done

Figure 4.1: The figure shows the flow of the presentation and metadata from the conference presenter to Zenodo. Firstly, the conference participant has to upload the presentation and metadata to Indico. Secondly, the user of this code can use Indico to obtain the metadata in the form of a set of JSON entries, on a webpage. The user also has to download the presentation pdf locally. Thirdly, using the Indico web address provided by the user of the code this code copies the metadata from the JSON file. Lastly, with the Zenodo access token provided by the user, this code uploads the metadata and the presentation to Zenodo.



with `requests.post`, which is also in the REST API. The last step is not implemented yet as we are still working on the Zenodo sandbox.

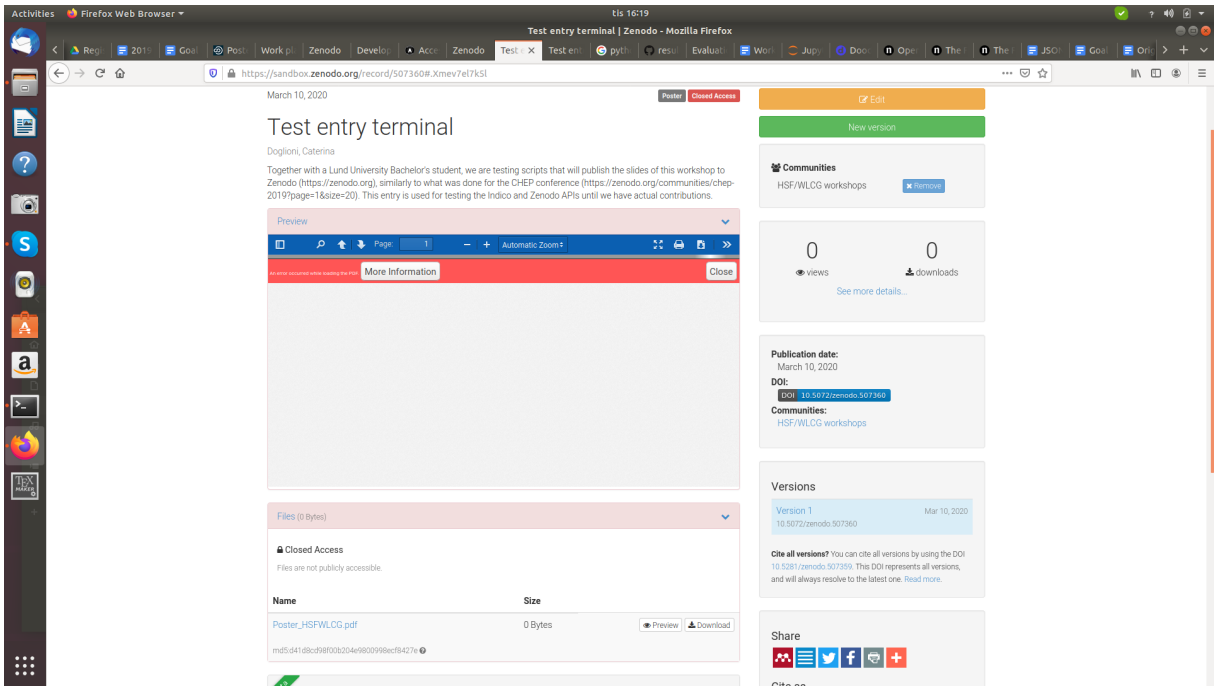
This code is implemented in a Jupyter notebook so that it is easier for users to understand each of the commands. For the time being, it takes a single test entry which is shown Chap. 3.3 and uploads it to Zenodo sandbox. The code can be found at `https://github.com/Christian-Cederholm/MyBachelorprojectcode`.

Issue with code validation The code exported all data to the Zenodo sandbox and no errors showed with the Zenodo REST API. However the image file in `sandbox.Zenodo` would not load and is empty. The issue is shown in Fig. 4.2.

I have written to Zenodo support, but they have not yet answered. For this reason, I have not yet validated the multi-entry support described below, and I have left this for future work.

Implementing multi-entry support Ultimately, it will be necessary for the code to be able to process more than one entry at a time, as a conference is made by many different contributions. To achieve this, I have started to split the code into three functions. The

Figure 4.2: The figure shows the issue with image files not showing



first function saves and stores the metadata, the second creates the Zenodo entry, inputs the metadata and publishes the entry and the last function imports the metadata from Indico and also calls on the first two functions. I have not yet tested this code, but it is also on Github for future users and developers to continue working on.

Chapter 5

Conclusions

In High energy physics, Indico is used as conference manager to host material about conferences, including conference contributions. This project was done since in Indico it is easy to access the pdf file of the presentation, but it is not easy to cite the presentation and its metadata. In this thesis, I described the first steps to upload the presentation and its metadata to Zenodo, a service that can host digital entries such as conference presentations. Zenodo gives each published entry a Digital Object Identifier along with its own webpage which shows title, description, publication date, authors, author affiliation, the presentation and the presentation title. The webpage also shows how many times the entry has been viewed and downloaded.

The work in this project was performed in 3 steps. First, I learned how to work with files and directories in the terminal, using the Python programming language in a Jupyter notebook. Secondly, I used the Zenodo REST API tutorial to write the base code which uploads a single entry to `sandbox.Zenodo`. Thirdly, I improved this code to make it easier to use and understand using a Jupyter notebook, and started the work towards making this code accept more than one entry.

The outcome of this project is a Jupyter notebook that shows the commands needed to upload one entry representing a conference contribution to the test Zenodo site, called `sandbox.Zenodo`. The improvements in progress are to turn the code into a function where the user can simply add in the variables, and splitting that function into three different functions that make the code easier to read and modify, and allow it to handle multiple entries. At the moment, this code can be used by Indico users who need to upload one entry to Zenodo. Once the improvements are completed, then it will be possible to add entire conferences to Zenodo.

5.1 Future work

The most important work is firstly to make it possible to upload multiple entries and multiple authors at the same time. This is vital as I currently do not know what happens if the code tries to handle multiple entries or authors. A small improvement that should have been in the code from the start is making the code download the date from Indico and use it as the publication date in Zenodo. But this is less important as the code still functions since Zenodo uses the current date if no other date is given. An improvement that is not merely making the code work properly is trying to make the code work even

if the JSON file is structured differently.

References

[1] Indico. https://getindico.io. Accessed: 2020-06-09.

[2] Zenodo. https://zenodo.org. Accessed: 2020-06-09.

[3] doi. https://www.doi.org. Accessed: 2020-06-09.

[4] Mönnich, A. *et al.* Indico 2.0 – the whole iceberg. *Journal of Physics: Conference Series* **898**, 102017 (2017). URL https://doi.org/10.1088/1742-6596/898/10/102017.

[5] OpenAIRE. https://explore.openaire.eu/. Accessed: 2020-06-09.

[6] REST API. https://developers.zenodo.org/#introduction. Accessed: 2020-06-09.

[7] Chen, X., Dallmeier-Tiessen, S., Dasler, R. & et al. Open is not enough. (Nature Phys 15, 113–119 (2019)). URL https://doi.org/10.1038/s41567-018-0342-2.

[8] Wilkinson, M., Dumontier, M., Aalbersberg, I. & et al. The FAIR Guiding Principles for scientific data management and stewardship. (Sci Data 3, 160018 (2016)). URL https://doi.org/10.1038/sdata.2016.18.

[9] [Provisional] HSF-WLCG 2020 Workshop. https://indico.cern.ch/event/867789/. Accessed: 2020-06-09.

[10] Jupyter. https://jupyter.org/. Accessed: 2020-06-09.

[11] Introducing JSON. https://www.json.org/json-en.html. Accessed: 2020-06-09.

[12] Rossum, G. V. & Drake, F. L. *Python 3 Reference Manual: (Python Documentation Manual Part 2)* (CreateSpace Independent Publishing Platform, 2009).

[13] Python Introduction. https://www.w3schools.com/python/python_intro.asp. Accessed: 2020-06-09.

[14] General Python FAQ. https://docs.python.org/3/faq/general.html. Accessed: 2020-06-09.

[15] The Jupyter Notebook. https://jupyter-notebook.readthedocs.io/en/latest/notebook.html. Accessed: 2020-06-09.

[16] Jupyter Notebook Shortcuts. https://towardsdatascience.com/jypyter-notebook-shortcuts-bf0101a98330. Accessed: 2020-06-09.

[17] What is REST. https://restfulapi.net. Accessed: 2020-06-09.

[18] Representational State Transfer (REST). https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm. Accessed: 2020-06-09.