FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Dendro Research Notebook: Interactive Scientific Visualizations for e-Science

Bruno Monteiro Marques

Master in Informatics and Computing Engineering

Orientador: João Miguel Rocha da Silva

Co-orientador: Tiago Nuno Mesquita Folgado Leitão Devezas

May 1, 2020

Dendro Research Notebook: Interactive Scientific Visualizations for e-Science

Bruno Monteiro Marques

Master in Informatics and Computing Engineering

May 1, 2020

Resumo

O conceito de Open Science trouxe ao centro das atenções da comunidade científica uma discussão sobre a reprodutibilidade no processo científico. A utilização de visualizações interativas como uma forma de apresentar dados de investigação, recentemente popularizada pelas tecnologias de *notebooks* computacionais, é vista como um passo inovador no sentido de obter uma maior reprodutibilidade.

Um dos requisitos da reprodutibilidade sobre resultados de investigação é a habilidade de analisar e reutilizar *datasets* científicos, tornando-se uma característica fundamental de um processo científico correto e transparente. Para atingir este fim os dados e os seus metadados devem ser descritos corretamente de forma a serem replicáveis e poderem ser reutilizados em diferentes contextos. Plataformas como o *Jupyter* ou *distill.pub* são bons exemplos de novas maneiras de comunicar cientificamente, permitindo aos utilizadores a construção de notebooks computacionais que contêm texto, visualizações interativas e código que pode ser visto e partilhado com outros utilizadores.

O conceito do Dendro Research Notebook nasce da vontade de capturar todos os componentes típicos do processo de investigação sobre a forma de um único documento interativo, enquanto simultaneamente apresentando visualizações descritivas de forma a facilitar o processo de interpretação dos dados.

O Dendro Research Notebook procura melhorar as capacidades de processamento, análise e reutilização dos dados tornando-se assim uma plataforma capaz de complementar os métodos tradicionais de depósito de metadados com a associação de visualizações e métodos de processamento sobre a forma de notebook.

O trabalho desenvolvido nesta dissertação teve como fundamento uma análise de conceitos do estado da arte como "Ciência Aberta" e "Dados Abertos", repositórios de dados, tecnologias de *notebook* e plataformas orientadas a publicação científica. Através de uma colaboração próxima com investigadores e utilizadores da plataforma este trabalho procurou estender as capacidades do Dendro de forma a integrar um serviço de *notebooks* computacionais web apto para criar, editar, partilhar e visualizar *Jupyter notebooks*.

Estas novas capacidades da plataforma Dendro foram demonstradas aos investigadores que colaboraram de forma próxima com o desenvolvimento desta dissertação, tendo através de um conjunto de sessões experimentais avaliando o sucesso da implementação. Obtendo resultados predominantemente positivos nos diversos aspetos da implementação analisados, compreendendo visualização, gestão de dados e reprodutibilidade.

Os resultados obtidos, em particular nos aspetos relacionados com a reprodutibilidade, demonstram a capacidade desta implementação em suportar um novo paradigma de investigação baseado na ciência aberta.

Keywords: Reprodutibilidade, Ciência Aberta, Dados Abertos, Partilha de Dados, Repositórios de Dados, Web Notebooks

Abstract

Open Science has contributed to bring reproducibility, the ability to reproduce research, to the center of discussion of scientific communities. By making available research datasets for others to reanalyse and reuse, scientists can thus contribute to increase the transparency of the scientific processes and workflows. To this end, metadata and data should be well-described so that they can be replicated and/or combined in different settings.

Innovative means to interpret existing scientific results and contribute to reproducibility have been developed in recent years. One such way is the combined usage of interactive visualizations and web notebook technologies. Platforms like Jupyter or *distill.pub* are examples of a new approach to communicate in science, allowing users to build computational notebooks that can contain text, interactive visualizations and code that can be shared and viewed by other users.

The concept of research notebooks stems from the idea that the ability to capture all typical components of a research study in a common interactive document form, while presenting interesting visualizations that assist in data interpretation, is an effective way to leverage the universal appeal of visual representations to improve reproducibility. Dendro Research Notebook aims to foster reproducibility by improving the processing, analyzing and reusing aspects of scientific data life cycle by allowing users to interact and share data processing methods within the Dendro platform via integrated web notebooks. The goal is therefore to make Dendro a platform capable of complementing traditional metadata records with visualization and data processing through the integration of notebooks.

The work developed in this dissertation was supported on the analysis of state of the art concepts like "Open Science" and "Open Data", data management platforms, computational notebooks technologies and publication-oriented platforms. By working closely with researchers and users of the platform, we extended the capabilities of Dendro with a computational web notebook service ready to handle creating, editing, persisting, sharing and visualization of Jupyter notebooks. These new features of the platform were assessed by several researchers that collaborated with this dissertation, through a series of experimental sessions evaluating the success of the implemented work.

The results obtained were predominantly positive in the evaluated dimensions of the implemented work, namely visualization, data management and reproducibility. The favorable results, particularly the ones related with reproducibility, indicate the ability of the developed work in supporting a new paradigm of scientific research based on Open Science. Its long term impact is however dependent on the acceptance, by the users of the platform, of the newly integrated tools to create, share and annotate their research with notebook-powered visualizations.

Keywords: Reproducibility, Open Science, Open Data, Data Sharing, Data Repositories, Web Notebooks

Acknowledgements

Before proceeding I want to acknowledge some people that made this work possible either through continuous advice, support and motivation. Firstly I have to thank both my supervisors João Miguel Rocha da Silva and Tiago Nuno Mesquita Folgado Leitão Devezas for their tireless dedication and patience towards me and this dissertation. The example set by their work ethic will have a long lasting effect in my perception of respect, thoroughness and persistence in their work and in the scientific method. I want thank my supervisors for always pushing and motivating me to excel, the best aspects of this dissertation are certainly a consequence of the effort put forward by them.

I also want to thank my family for always supporting my decisions devoting so much time and effort to provide me with the opportunity to study and dedicate myself to my passions. I certainly wouldn't be able to complete this work without their support. To my mother and my father through their example of perseverance and determination that always motivated me to reach further, thank you.

Finally I would also like to thank my friends, the family I built during these years I spent in university, I could not have been blessed to have better companions than you. I can't imagine ever achieving what I have today if not for the belief we have in each other. I truly feel like under many circumstances I carried onward thanks to your immense support alone and I hope only to be able to repay that one day. The unimaginable amount of encouragement we provided both through the best and worst moments of these years will one day seem trivial, but I know the memory of what you have done for me, I will carry for a lifetime, thank you.

Bruno Marques

"An expert is a person who has found out by his own painful experience all the mistakes that one can make in a very narrow field."

Niels Bohr

Contents

List of Figures

List of Tables

Abbreviations

RDM	Research Data Management
OA	Open Access
US	United States
CSV	Comma Separated Values
INESC TEC	Instituto de Engenharia de Sistemas e Computadores, Tecnologia e Ciência
API	Aplication Programming Interface
ipynb	IPython Notebook file extension
URL	Uniform Resource Locator
FEUP	Faculty of Engineering of the University of Porto

Chapter 1

Introduction

Computational notebooks are virtual programming environments that often combine the abilities of a word processing software with a kernel and shell of a respective programming language. These computational notebooks, sometimes designated as notebook interfaces, find usage amongst publishers, data analysts, universities and others, including personal use [41].

This concept was firstly introduced as the Wolfram Mathematica in 1988. It featured a notebook like graphical interface combined with a processing kernel capable of executing the Wolfram language [29]. As time went on a number of different kernels for different languages were introduced ranging from MATLAB [1], Pyhton [2], Julia [3], among others [54]. In recent years, notebooks have found utility by providing interactive ways to store data in a easily presentable condition, bridging the gap between storing data, annotation and presentation. The ability to execute code within notebook environments creates an opportunity for users looking for an uncomplicated way to build their code, making computational notebooks an attractive tool for researchers. These computational notebooks' ability to retain data alongside processing code and visual representations for future analysis and distribution create a environment for reproducibility [28].

Data visualization on the other hand has for long played an important role in exploration, analysis and presentation of scientific data [52]. Producing visualizations to present research findings has always been an integral step of the scientific process through advances in graphics hardware, combined with the development of new visualization methods, techniques and systems which have provided new ways to interpret scientific research [50, 38]. Widgets allowing scientists to execute and aggregate visualizations based on the code processing data all captured within the notebook environment push towards a paradigm on scientific data being presented via interactive visualizations. These visualizations can offer several benefits to conventional presentation such as the refining of parameters/attributes, object manipulation tasks (selection, translation, rotation, scaling, aggregation and time), filtering of data sets, among others [38, 28].

Finally the term e-Science dates back to 1999 referring to computationally intensive science that is carried out in highly distributed network environments using large data sets [4]. Since the

[1] mathworks.com
[2] ipython.org/
[3] github.com/JuliaLang/IJulia.jl

term's inception that global scientific collaboration has evolved to take on many forms, however from the various initiatives around the world, a consensus is emerging: research collaboration should aim to be "open" or at least there should be a substantial measure of "open access" to the data and information underlying published research [13].

In the interest of supporting this paradigm shift towards a more Open Science it is important to provide researchers and other interested counterparts tools that are capable of accommodating this change. A scenario where most of the steps of the scientific research are encapsulated in a convenient platform supporting collaboration and sharing is ideal in order to support this effort.

1.1 Context

This increased prevalence of Open Science has brought reproducibility to the center of discussion of the scientific community. The ability to reproduce scientific research is a requirement for ensuring the transparency and correctness of the research workflow. This focus on reproducibility can be seen through various publications, platforms, technologies and other elements of the scientific process [9, 45].

The Dendro platform is a scientific data management platform that has been under development at FEUP. It aims to facilitate the management of data sets, documents and other digital materials produced by research groups. Dendro can extract and index contents from files in a seamless manner, allowing users to upload files in different formats like CSV or Excel, and automatically store them in a Big Data database suited to be queried [12]. This platform seeks to provide its users with effective tools that allow them to pursue these new paradigms in scientific research looking to associate its methods of storing data with solutions to increase reproducibility. The usage of interactive visualizations heightened by the expansion of the notebook interface technology can be seen in platforms like distill.pub and the popular Jupyter Notebook. They are good examples of emerging technologies associated with reproducibility and by effect Open Science [14, 38].

This was the origin to the idea behind this dissertation: combining the strong points of notebook interfaces, in particular their flexibility in providing interactive visualizations, with Dendro's data storing environment, empowering its users with a tool that allows for depositing data, executing code over that data and create visualizations, therefore increasing the appeal for reproducibility.

To understand the impact that the inclusion of computational notebooks can have in a platform like Dendro we must discuss the most prominent and widely used notebook interface at this time. The Jupyter technology is an open-source software for interactive computing that supports dozens of programming languages, which is one of the factors that makes it the most attractive for the scientific community. The support for this platform is extensive, including its dedicated and large user base and open-source software features. Its multi-language support also makes it a flexible option for different groups of users [26].

Jupyter technology allows scientists to create their own dynamic notebooks, with prose, interactive visualizations even code snippets that can be shared and viewed by other subjects. Distill.pub is an open access scientific journal that uses these tools to expand the way scientific articles are written [14, 38].

1.2 Motivation and Goals

This dissertation looks to bring these aspects together in Dendro, trying to elevate the platform to a standard that will fulfill the necessities of the most updated guidelines for reproducibility and Open Science [53, 41, 35].

With this proposal we will be able to observe how the ability to pair data and processing methodologies with annotations and or visualizations will increase the value of said data appealing to reproducibility.

This endeavour looks to bring Notebook Interfaces to the Dendro platform setting out to further develop it into a powerful research tool.

With "Dendro Research Notebook" the deposited data in Dendro will integrate the strengths of both the notebook and the Dendro platform, combining the visualization and dynamic interactions of notebook platforms with the already established data management tools at Dendro. This leads us to the hypotheses:

> The integration of sharing, processing and visualization aspects of computational notebooks in data management platforms will improve reproducibility and encourage Open Science.

In order to fulfil this hypothesis, this work sets out with goals mainly focused on inciting reproducibility. Dendro as a research notebook should allow for the creation of notebooks from the platform, sharing file and data structure directly to the notebook. The way data is described within Dendro will be expanded upon by the notebook, allowing researchers to describe their works with visualizations, executable code and text. Researchers will also be able to easily find other notebooks within the public repositories in the platform in order to facilitate reuse of scientific data. Interactive visualizations alongside snippets of the notebook will be integrated with the data and existing project descriptions. Researchers will also be able to download and edit metadata belonging to the notebook.

To perform this implementation, consultation with researchers will be crucial in order to pinpoint their necessities. Finally, to assess the value of the implementation, various evaluation tasks will be performed in order to assess some aspects of usability, like effectiveness, efficiency and satisfaction. This work will demonstrate how the visualization solutions and interactivity with scientific data brings incentive to reusability, contributing for a better data management workflow in the future.

We hope that this dissertation will contribute to the research data management field by successfully modelling a new workflow where data is closely connected to its presentation and processing, concentrating several steps of the scientific research in the same tool and keeping them closely linked will emphasize the importance of bridging the gap between data and visualization. This will develop a bigger concern with the reproducibility of data and generally improve these aspects within Dendro.

1.3 Document Structure

This document in addition to the introductory chapter, has a literature review, a methodological approach chapter, an experiments and outcomes chapter, and a conclusions section.

The literature review section is divided into three parts. In order to understand the motivation behind this work we must analyse how Open Science has come to the attention of the scientific community and what are its current requirements. We should also have a brief overview of current notebook technologies in order to correctly assess which technology to implement on the platform, as well as study visualization and its contribute to data reproducibility.

Starting with Open Science and Open Data, we will look to define them and their obstacles, and also explore the data life cycle aspects, FAIR principals, web notebooks, and their interactions with Open Science, reproducibility and data managements platforms. Finally we will overview visualization under an Open Science context and its possible impacts.

In the methodology approach section, we will go over all of the implementation process from problem formulation, requirements inquiry and implementation. Starting with requirements gathering, we will discuss the problem formulation, use cases and technology selection. We will then present a structural modeling of the implementation and describe the implementation process in detail.

In the experiments and results sections, we will discuss both the experimental procedure that validated this dissertation's work and the satisfaction with the implemented solution. We start with an analysis of the implementation outcomes, describing all of the implemented features and measuring the degree of success of the implementation. We then present an experimental analysis over these features in order to better evaluate the success this dissertation achieved from the end users' standpoint.

Finally, the last chapter overviews all of the performed work in this dissertation, its impact, and possibilities for expansion in future work.

Chapter 2

Literature Review

This section of the dissertation addresses the current state of the art of the elements necessary to understand this work. These range from different fields, so a careful approach will be used in order to understand the different aspects of these fields that combine into creating the solution this work came to. There are three main aspects that should be understood to grasp the concepts fundamental to this dissertation: Open Science, web notebooks and visualization. It is essential to understand the impact that open science has had on the scientific community especially in order to understand the motivation behind this thesis. Web notebooks should be overviewed in order to understand the possibilities these platforms provide and determine which technologies are right for this dissertation's work. Finally data visualization is an integral part of this dissertation and one of the main aspects in which this work can push the Dendro platform to an effective reproducible research tool and therefore an analysis of the visualization solutions that integrate into these notebooks is necessary.

2.1 Open Science and Open Data

e-Science is a term dating back to 1999 first used by John Taylor, director of the UK's science and technology office at the time. It refers to computationally intensive science that is carried out in highly distributed network environments, using immense data sets that require grid computing [4]. Global scientific collaboration takes many forms, but from the various initiatives around the world, a consensus is emerging: research collaboration should aim to be "open" or at least there should be a substantial measure of "open access" to the data and information underlying published research [13].

Open Science represents a new approach to the scientific process based on cooperative work and new ways of diffusing knowledge by using digital technologies and new collaborative tools [35]. For researchers, however, this translates into a need to adapt various steps of the scientific process. The principles of openness should be extended to the whole research cycle, from data collection, processing, storing, preservation, distribution, reuse and even from hypothesis composition, as shown in Figure 2.1.

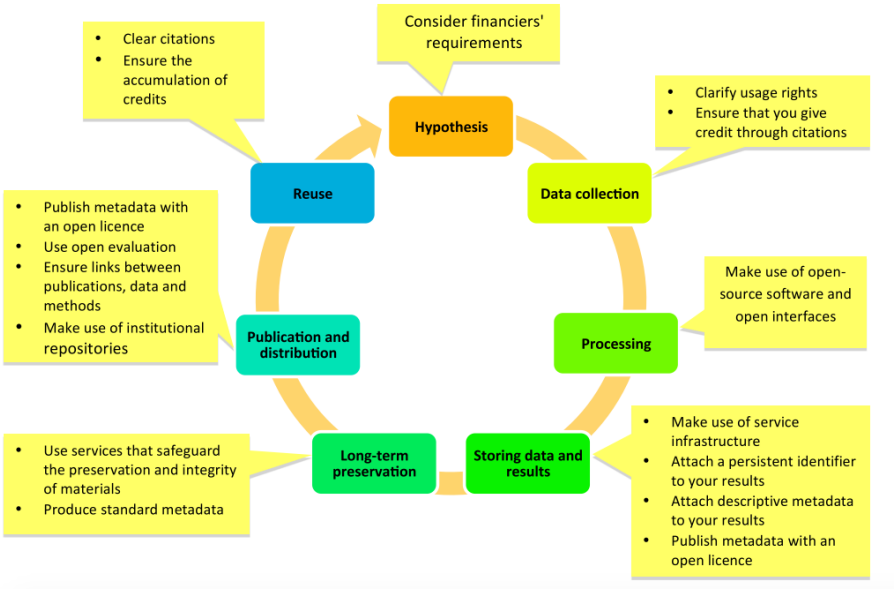

Figure 2.1: Promoting openness at different stages of the research process [35].

Open Science Monitor [1] analyses data and case studies covering access to scientific publications. It divides publications into "Gold Open Access" and "Green Open Access". In Gold OA (Open Acess), research outputs are made available under an open access license by the publisher on the journal website. Under Green OA terms, research outputs are not made available by the publisher, but by the author(s), who independently deposit data and publications in an open access repository [9]. It should be noted however that "Gold Open Access" is commonly associated with an Article Processing Charge whose value can reach five thousand dollars in some journals. This stands as a clear obstacle to a broader existence of Gold Open Access as a standard for scientific publishing [47].

In Figure 2.2 we can see that there has been a general increase in research work made open access by the publisher or publication. We can also denote from the figure how Gold Open Access is gaining popularity over time. This can be seen as a positive since it removes responsibility from the author to make the research available — this is a step in the right direction since plenty of projects aim to improve in ways to make their research available. If this were to become the norm for a majority of projects we could see a big improvement in open science indicators across research communities [31, 53, 35].

In addition to the increase in Open Access publication we can also see in Figure 2.3 which countries lead the way in this field. This is not just a scientific trend since countries like the Netherlands are trying to develop strategies to achieve 100% open access by 2020 [31].

[1] fosteropenscience.eu/content/open-science-monitor

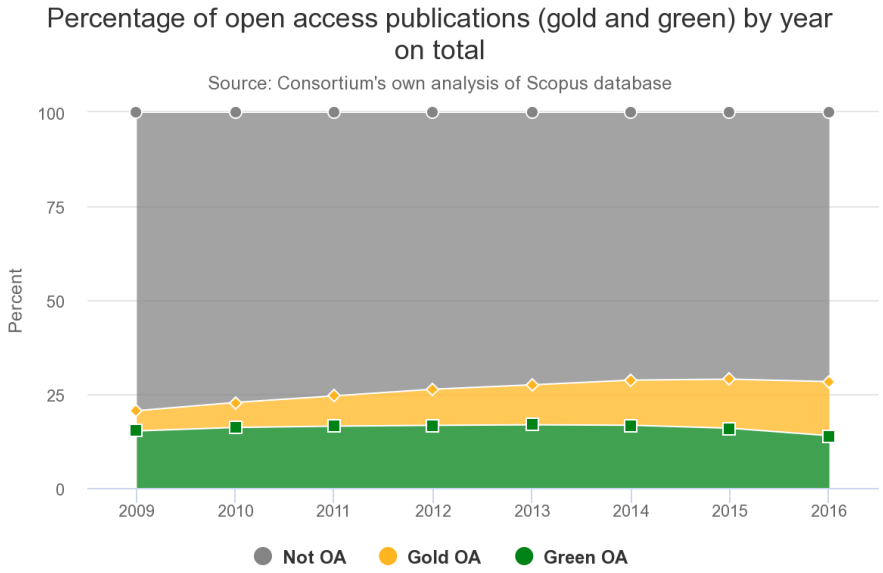

Figure 2.2: Relative analysis of total number of Open Access publications per year [10].

2.1.1 FOSTER Open Science Training Tools

FOSTER is an EU project aimed at identifying, enriching and providing training content on relevant Open Science topics in support of the European Commission‘s Open Science Agenda in the European Research Area [35]. This project focuses on providing Open Science training to the European research community with methods such as supporting young researchers in their compliance with open access policies and looking to integrate OA principles in current research workflow, while strengthening the institutional training capacity beyond the project.

In Figure 2.4 we can distinctly see how Open Science comprises several fields of work. Ranging from reproducible research, evaluation, tools or even policies. Open Data in particular is relevant to this work since we plan to integrate these concepts with scientific data management. However Open Science does come with some challenges.

2.1.2 Barriers to Open Science

In a recent case study of the Netherlands' Plan on Open Science, a particularly interesting topic were the barriers encountered [31]. One of the most relevant was the storing and sharing of data since discipline-specific data protocols within technical and policy-based preconditions are needed to ensure a consistent and FAIR access to research data [53]. These privacy issues, proprietary aspects, and ethics are common barriers in open science to all fields, and data management platforms are trying to provide solutions to create solid infrastructure to support the reuse of scholarly data.

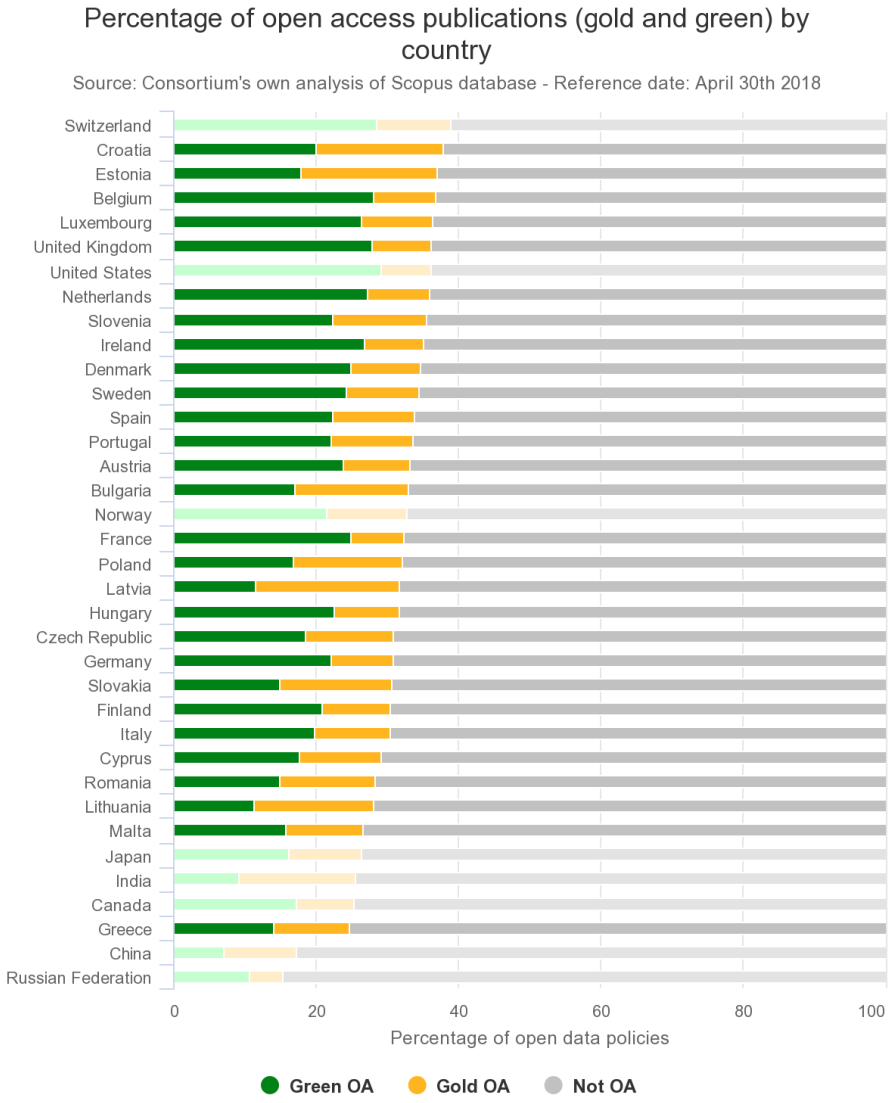

Figure 2.3: Relative analysis of total number of Open Access publications per country [10].

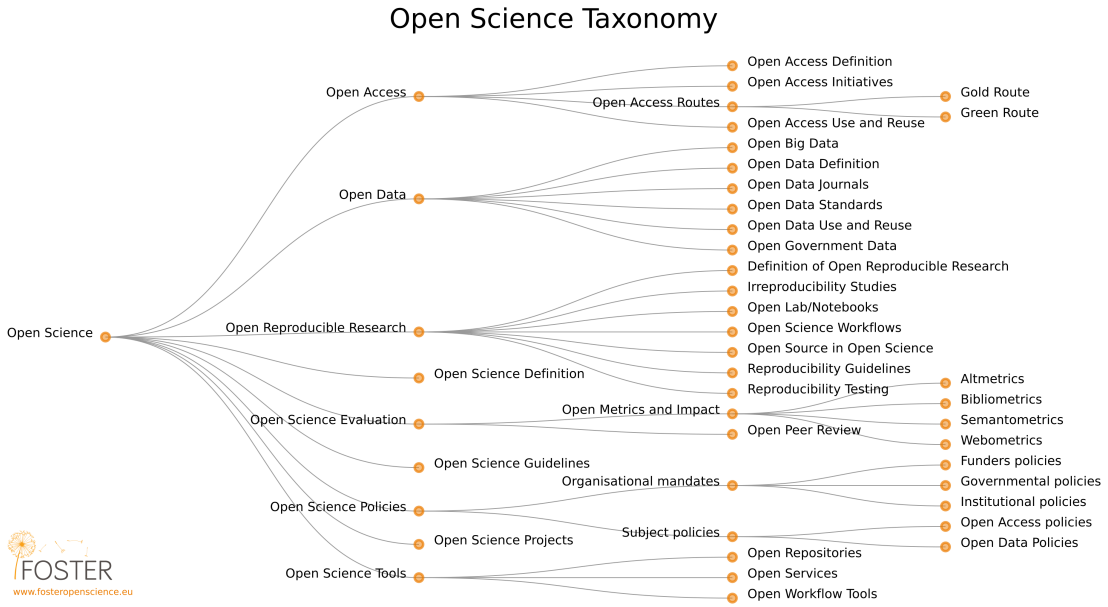

Figure 2.4: Open Science taxonomy [35].

2.1.3 The FAIR Guiding Principles for Scientific Data Management

A diverse set of stakeholders from academia, industry, funding agencies, and scholarly publishers have come together to design and jointly endorse a concise and measurable set of principles that we refer to as the FAIR Data Principles, with the intent for these to act as a guideline for those wishing to enhance the reusability of their data holdings [31, 53]. These principles focus on four mains aspects: findability, accessibility, interoperability and reusability. The findability of data consists of making sure that data is easy to find by humans and computers, allowing for automatic discovery of datasets and services crafted with FAIR in mind. Accessibility consists of ensuring that the assigned data provides a clear path on how it can be accessed. Interoperability is also fundamental considering the necessity to interoperate data between different workflows, tools, processing methods and storage. And finally, the ultimate goal is the optimization of the reuse of data, so steps like making sure data is well described in order to be reused are vital [19].

2.1.3.1 Data Life Cycle

The data life cycle shown in Figure 2.5 is the sequence of stages that a particular unit of data goes through from its initial generation or capture to its eventual archival and/or deletion at the end of its useful life [40].

This is one of the cornerstones of scientific data management: respecting this process and finding ways to optimize each segment of the cycle has been on the core of the developed work in this area. However, researchers believe there is a “reproducibility crisis”, since the complexity and extent of scientific experiments and data collections has gone so far that reproducing experiments and reusing scientific data from other research groups is becoming increasingly difficult [2].

Figure 2.5: Data Life Cycle [40].

Considering the data life cycle diagram in Figure 2.5 we can begin to understand in which steps of the cycle this dissertation work can affect reproducibility in a positive way. The integration of web notebooks into the workflow can affect almost every step of this cycle one way or another. However when considering the core function of the notebook, storing, processing and analysis methods, making them easily interchangeable alongside data itself, cataloguing data and experiments within notebooks allowing for an interactive way to preserve data, we can see just how impactful the web notebook can be. Since increasing reproducibility is one of the core aspects of this work, the employment of web notebooks' abilities over the data life cycle opens new exciting possibilities in this field [38, 54].

2.1.4 Scientific Data Management Platforms

With the growing number of scholarly papers being published and a growing awareness of the importance, diversity and complexity of data generated in research context, platforms look to offer solutions designed for managing research data [30]. These solutions are being actively developed by both open-source communities and data management-related companies focusing on description and long-term preservation of data [30, 1]. These require detailed, domain-specific descriptions to be correctly interpreted [39].

2.1.5 Data Management Platform Requirements

Data management platforms provide a variety of functions while maintaining description and long-term preservation as the core focus of their functionality. Different platforms bring distinct solutions packages and this variety is what sometimes creates difficulty selecting an appropriate management tool [1]. In order to find impactful ways this work can affect RDM (Research Data

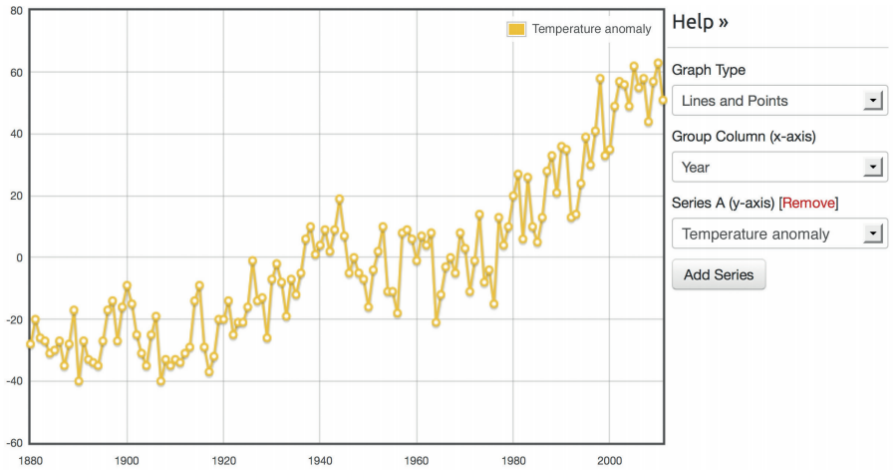

Figure 2.6: CKAN visualization example provided by Recline Data Viewer [8].

Management) we should study reference RDM platforms and the features they display in the context of open access. And analyse to which extent the integration of a notebook will cover the existing features these platforms have to offer. The most important solutions in the same context as Dendro[2] are instances running at both research and government institutions [1]. Platforms like DSpace[3], CKAN[4], Zenodo[5], Figshare[6], ePrints[7] and EUDAT[8] fill this category and all comprise some sort of previewing of data. We will look at visualization in particular since promoting interactivity and allowing researchers to annotate their data with visualizations is one of the focuses of this work.

2.1.5.1 Visualization in Scientific Data Management Platforms

Even after a brief inspection it is quite easy to see CKAN as one of the most prominent portal software framework used for publishing Open Data, used by several governmental portals [32]. Intending to maximise re-use of data, quite in line with the work we propose, CKAN's datastore can store structured data and provide access to it via an API. This will simplify the process for researchers checking and re-using data from earlier research. This is one of the aspects where CKAN shines the most by creating interactive data visualizations, using the built-in Recline data viewer.

Visualizations also include map plots of geo-coded data or image files displayed on their resource pages. Even though these visualizations are a step in the right direction and retain value in their convenience to researchers, they still don't provide the ability to manipulate data or generate

[2] https://github.com/feup-infolab/dendro
[3] https://duraspace.org/dspace/
[4] https://ckan.org/
[5] https://zenodo.org/
[6] https://figshare.com/
[7] https://www.eprints.org/uk/
[8] https://www.eudat.eu/

custom visualizations the same way a web notebook would. The best of both worlds would be to have some simple automated visualization aspects just the way CKAN provides, while having the opportunity for users who would want to go more in depth to develop their own visualizations and manipulate and test over the research data in question in an accessible way.

2.1.6 Dendro

Dendro is a platform with "Dropbox like" capabilities and extended description features. It works as a data storage and description platform that was designed to help researchers and users describing their data files. Dendro is also collaborative and is designed to supports users collecting and describing data, with its roots based on the field of research data management [12].

Dendro is designed to support this collaborative work including features such as metadata versioning, permission management, editing and rollback and public, private or metadata only visibility. This makes Dendro a flexible framework for data description. The inclusion of a web notebook service within this platform can bring advantages, however to understand this we must look at how web notebook can contribute to research data management.

2.2 Web Notebooks for Open Science

Notebook interfaces were first introduced in a very rudimentary way around 1988 in the Wolfram Mathematica 1.0 software for the Macintosh [29]. A notebook interface is a virtual environment used for programming which pairs the functionality of a word processing software with a shell and kernel programming language. As scientific work becomes more computational and data intensive, research processes and results become more difficult to interpret and reproduce [54].

Web notebooks are a type of web based notebook interface that combines the features of the notebook with the accessibility of web platforms, creating exciting new opportunities for open science. The value of notebook interfaces comes from their ability to capture all typical components of a research study in a common, interactive and document-like form. Notebooks can also be displayed in a slide show mode for interaction with decision makers [54]. Therefore, web notebooks can be seen as an effective way to manage and exchange knowledge among scientific communities.

2.2.1 State of the Art in Web Notebooks

In order to understand current versatility and capabilities of notebooks we analyze the most successful notebook technologies on the market. Since its genesis in 1988, notebooks have taken different forms, evolving alongside recent trends in computer science. From the first iterations of Wolfram Mathematica[9] to cloud based systems like Microsoft's Azure Notebooks[10] or Google Colaboratory[11], these tools provide different services and have different strong points. Jupyter Note-

[9] https://www.wolfram.com/mathematica/quick-revision-history.html
[10] https://notebooks.azure.com/
[11] https://colab.research.google.com

book had its origin in the IPython notebook[12] and after enjoying success is now evolving towards a new revised and more versatile platform JupyterLab[13]. It also integrates several popular forthcoming platforms like Google Colaboratory project, so the future looks bright for Jupyter [28].

Table 2.1: A comparison of several popular Research Notebook frameworks

	Developer	Language Support	Environment	Multi-User	Visualization	License	Cost
Jupyter Notebook	Jupyter	Multi-Language	Local	No	ipywidgets	BSD	Free
JupyterLab	Jupyter	Multi-Language	Local	No	ipywidgets	BSD	Free
JupyterHub	Jupyter	Multi-Language	Cloud	Yes	ipywidgets	BSD	Free
Jupyo	Bits and Bots LLC.	Python, R, and Julia	Cloud	Yes	ipywidgets	Proprietary	Subscription
Google Collaboratory	Google	Python	Cloud	Yes	matplotlib	CC	Free
Observable	Observable Inc.	JavaScript	Web	No	Vega and D3	MIT	Free
Azure Notebooks	Microsoft	Python, R, F#, etc.	Cloud	No	ipywidgets	MVL	Variable
R Notebooks	R Studio Inc.	R, Python, Bash, etc.	Local	No	r2d3	GPL	Free

In the comparison shown in Table 2.1 we take a look at some of the current popular notebook technologies. Many of the options in the market today are Jupyter based like Jupyo or Azure Notebooks, and present alternatives for multi-user cloud running that allows users to share and work in their notebooks simultaneously without worrying about server setup or maintenance. The Google Colab initiative intends to bring these features to its users for free, however it is more limited in Language Support and it is highly focused on Machine Learning projects. JupyterHub is a multi-user version of the notebook designed for companies, classrooms and research labs that aims to be customizable, flexible and highly scalable; it is also the closest available multi-user experience of the original Jupyter Notebook. Regarding the visualization capabilities of each platform, on the visualization column we point out some of the more interesting alternatives each platform provides for data visualization.

Some of the more feature-rich solutions are ipywidgets[14], which allows the creation user interface controls for exploring code and data interactively, and therefore allow for a extensively community support basis for Jupyter Notebooks in this field. R2d3[15] for R Notebooks also provides powerful publication focused visualization solutions for the users of the platform. Finally, Vega, a visualization grammar commonly used in Observable notebooks allows the creation, saving and sharing of visualization designs in a JSON format, to generate web-based views using Canvas or SVG combining great versatility with aesthetically pleasant visualizations.

2.2.2 Popularity of Notebook Technologies

An interesting exercise could be to evaluate the popularity of Notebook solutions in order to try to understand which technologies are most widely used and get bigger community support. Besides the previous comparison of technologies based on their technical support, we can also try to obtain some metrics on the applicability of a notebook technology based on their popularity rating. So, through a comparison based on GitHub metrics seen in Figure 2.7, we can extract

[12] https://ipython.org/
[13] https://jupyterlab.readthedocs.io/en/stable/
[14] https://ipywidgets.readthedocs.io/en/stable/
[15] https://blog.rstudio.com/2018/10/05/r2d3-r-interface-to-d3-visualizations/

	Contributors	Watchers	Forks	Stars	Popularity Score ▼	Standard Deviation	Popularity Rating
Jupyter	370	274	2330	5347	67427	$\sigma > 1$	Very Popular
SageMath	435	119	262	916	45066	$\sigma > 0$	Popular
JupyterLab	209	329	981	7229	42429	$\sigma > 0$	Popular
JupyterHub	131	266	992	4204	31234	$\sigma > 0$	Popular
R markdown	73	135	622	1301	16791	$\sigma < 0$	Unpopular
Databricks	23	19	56	76	3086	$\sigma < 0$	Unpopular
Azure Notebooks	15	49	44	207	2977	$\sigma < 0$	Unpopular
Google Colab	6	33	55	320	2070	$\sigma < 0$	Unpopular
Observable	4	19	17	281	1191	$\sigma < -1$	Unpopular

Figure 2.7: Notebook technologies comparison based on Github metrics [28].

some deductions relative to the current state of notebooks adoption. This comparison was made through an analysis of the different official repositories for the notebooks, where the number of contributors, watchers, forks and stars were utilized in order to establish which of the technologies had a more active support.

Beyond the presence of well established technologies like SageMath or R Markdown we can definitively see a predominance of Jupyter based technologies. Besides Jupyter itself, both JupyterLab and JupyterHub appear as some of the most popular repositories, making Jupyter a definite leader when comparing developing notebook technologies. Developers wishing to implement notebook solutions in their work, or simply researchers looking for a tool to accommodate their workflow should always consider a large array of options and strive for interchangeable methods in order to promote reproducibility. However if a choice must be made it is quite clear that Jupyter Notebook is a leading options in this field [28].

2.2.3 Jupyter Notebooks

The Jupyter Notebook is an open-source web application whose uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more [26]. These notebooks can be shared as executable files, using the Jupyter Notebook Viewer, or via third party execution/management environments such as Anaconda Enterprise Notebooks [38]. In addition, JupyterHub is able to create a multi-user Hub which spawns, manages, and proxies multiple instances of the single-user Jupyter notebook server[16] [35]. Due to its flexibility and customization options, JupyterHub can be used to serve notebooks to a class of students, a corporate knowledge base, or a scientific research group. Notebooks can produce rich media output that combines narrative text, static images, code, and dynamic, interactive output produced by Jupyter interactive widgets. These widgets allow the presenter to execute and visualize results in real time based on the code contained in the notebook. In addition to the standard notebook layout, Jupyter notebook cells can also be displayed in a PowerPoint presentation style mode [35].

[16] https://jupyter.org/hub

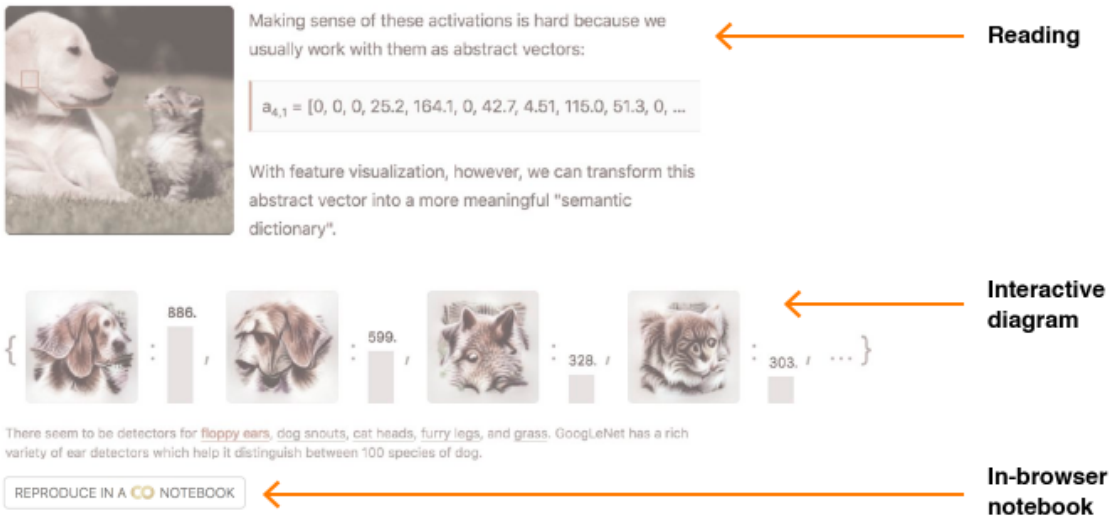

Figure 2.8: Example of paper at Distill.pub [18].

2.2.4 Distill.Pub

Distill.Pub [14] is an interesting platform to analyze, as it started off as an open-access scientific journal aimed at improving communication of machine learning results [34]. This format shares several similarities with this work's goals and therefore by considering some of the aspects of Distill we can try to understand what generates its popularity and which aspects compose the appeal of this platform [18].

Firstly, the editors behind Distill inferred that the platform had over a million unique readers, and more than 2.9 million views. Distill papers have been cited 23 times on average, placing Distill in the top 2 % of academic journals indexed by Journal Citation Reports[17]. However, it's also important to remember that Distill is a very small publication selecting very few papers, as stated by the editors themselves [18]. The three major areas where Distill's concept have a major impact are: interface, engagement and software engineering practices. To quote the authors on their interface, "Bolstered by the interactivity, it invites readers to step into a way of thinking.". This tries to reflect on how upon interacting with the publication itself readers perceive concepts in a different manner that can stimulate new ways of looking at the subject matter by rethinking it. Engagement wise there's an improvement derived from reading a paper, to testing and building on it. With Distill however we see papers where engagement is a continuous spectrum, composed by reading, interacting and even being able to reproduce the notebook, as seen in Figure 2.8.

Finally, on the software engineering end, every Distill article is housed within a GitHub repository, and peer review is conducted through the issue tracker giving readers greater transparency into the publication process.

[17] https://distill.pub/2018/editorial-update/

2.2.5 Advantages for Researchers

The process of describing the experimental methodology with enough detail as to enable data sharing and foster result analysis can be tedious. However if an experiment is not well described, potential re-users and even the creators of data could be unable reproduce prior results. It is often so much so that it becomes more cost-effective for researchers to try to reproduce data than to reproduce previous research products [2]. Research Notebooks can help to cope with these issues, as they serve as a means to directly share the process in a platform that can reproduce it immediately, while still storing metadata and presenting visualizations over the actual data. This allows other researchers to perform direct and on-demand analysis on the data, running the code as the original creator built it, therefore improving the ability of researchers to reproduce the experiments of others and execute their research work in a much simpler and secure way.

2.2.5.1 Sharing Data Increases Citations

Sharing data requires describing data to make it usable by other researchers, and this is a time consuming process, so if there is no direct mandate, e.g. from a funding agency, other strong incentives need to be in place to convince researchers to invest the time needed. This incentive could come from a citation advantage. It has been found, through bibliometric analyses, that a citation advantage for astrophysical papers in core journals exists when the works are associated with data by bibliographical links [17]. These papers receive on average significantly more citations per paper per year than papers not associated with links to data.

2.3 Data Visualization in Open Science

When talking about visualization it is important to recognize that this is a mature field with extensive work and a multitude of applications. A broadly explored concept whose aspects are commonly considered when looking to build upon developing ideas. This ever evolving field has mutated and adapted to the advancements around it time and time. In this new context for open science however the values of visualization are once again reiterated and expanded upon in light of reproducibility. To understand this we must go over some fundamental values of visualization [50].

2.3.1 What is Data Visualization?

Data visualization has always played an important role in exploring, analyzing, and presenting scientific data [52]. We can define it as a set of techniques used in order to create imagery that aims to facilitate the communication of a concept or message, which can take the form of graphs, diagrams, images, animations among others. Through mapping attributes to visual properties like position, size, shape and colour, visualization designers leverage perceptual skill in order to help discern and interpret patterns within the data, providing a powerful method to make sense of data [22].

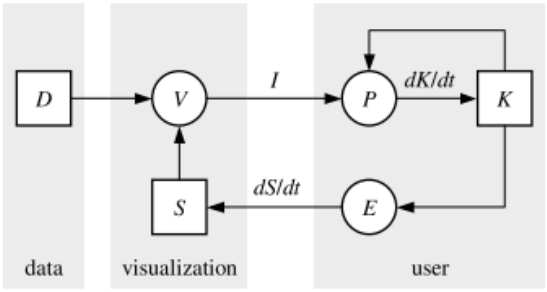

Figure 2.9: Value of visualization scheme proposed by [50].

2.3.2 The Value of Data Visualization

Finding value within a visualization should be an integral part of the scientific process. Producing good visualizations should not just be another step in the scientific process, but a step towards clearing the way for analysis and re-usage of data. Much progress has been made in standardizing processes towards creating good visualizations, the advances in graphics hardware have been great, and many new methods, techniques, and systems have been developed [50].

Some models like the one in Figure 2.9 display how Data D through a specification S is translated to image $I(t)$. From the users' perception P, knowledge K is obtained over time in different degrees depending on the user's original knowledge. Many visualizations, especially in recent years, push towards incorporating interactivity adding a new value of exploration E, allowing for the user to adapt the specification in order to facilitate the knowledge retrieval process. This data can be from spreadsheet data to the text of novels, but much of it can be represented as variables, arrays or transformed (perhaps with loss of information) into this form. Then we must be able to transform this data into useful graphical visualizations that allow ultimately the user to extract appropriate conclusions [7]. This is especially important when we talk about scientific data, since its sensitivity to any distortion caused by the transformation from raw data to processable data while paired with an unappropriated viewing method can lead to bias [23].

A valuable visualization has a high knowledge and is able to deliver information easily and extensively to the user, so it's quite clear how this can be appealing to scientists publishing their works, analysts trying to interpret data, etc. In [6], major ways in which cognition can be amplified by visualization are proposed. Even though this work is somewhat dated, some of these proposals still carry relevancy when analysing them against the new trends brought by web notebook technologies. The need to search for information in data is largely cut if good visualizations are in place. Visual representations are vital ways to enhance the search for patterns in data, encoding information in manipulable mediums and using perceptual attention mechanisms for monitoring data. This is usually associated to visualization solutions that are greatly amplified by web notebook's abilities.

2.3.3 Data Visualization in Open Science

In order to understand how visualization fits in the context of Open Science we must also understand how it leverages some of the best aspects of the notebook. Some existing work shares a very positive outlook on how notebooks can contribute towards open science [38, 54, 37]. Inspecting data is one of the drivers for researchers to read publications and share their own, and many funding agencies and journals alike require the release of data as a condition for funding or publication [38]. This shared data is usually paired with visualizations in order to take advantage of the aspects we have seen. These figures are however typically rendered as static images, and divorced from the underlying data, preventing readers from exploring them fully using tools like segmentation or zooming on features of interest [37].

2.3.3.1 Visualization Solutions for Open Science

Visualizations plays a role in several applications concerned with advancing towards Open Science. From data management platforms to notebooks themselves all of these aggregate the possibility to pair data with description. Visualization empowers users with ways to explore and interact with data streamlining engagement with scientific research [28]. This brings us back to web notebooks as a tool for open science, through their ability to produce a rich media output combining narrative text, static images, interactive visualizations, and code running over data dynamically [54]. The inclusion of interactive visualizations can be used as an indicator of maturity of a scientific research dataset according to the FAIR principles [53]. One of the proposed methods for the evaluation of level of reproducibility of scientific research is [28]:

- Level 0 – Scientific research with no associated Data or data under non-Open Access licensing;
- Level 1 – Research data in Open Data but no bundled transformation steps;
- Level 2 – Research data in Open Data and bundled transformation steps executable on-demand;
- Level 3 – Research data with Open Access with transformations and interactive visualizations.

Being aware of visualization's incentives, and how visualization plays a major role within the notebook format we should get familiar with state of the art visualization solutions. In a comparison based on GitHub metrics seen in Figure 2.10, it is compelling to evaluate which visualization libraries are most popular among users, as a means to understand which technologies would be relevant for the integration with this work. It makes sense to analyze some of these popular libraries in a more in-depth manner to further assess which would be essential to this implementation. Vega [42] and D3 [11] are well established visualization solutions that various notebook

	Contributors	Watchers	Forks	Stars	Popularity Score ▼	Standard Deviation	Popularity Rating
D3	123	4026	20373	82320	377640	σ > 4	Extremely Popular
matplotlib	787	533	3964	8685	129815	σ > 4	Extremely Popular
bokeh	346	398	2395	8962	72012	σ > 2	Very Popular
ggplot2	173	325	1357	3607	39247	σ < 0	Popular
Shiny R studio	42	335	1459	3254	28324	σ < -2	Popular
Seaborn	87	235	904	5720	27290	σ < -2	Popular
Vega	67	295	714	6631	25701	σ < -2	Popular
pyecharts	16	241	1008	4694	21034	σ < -2	Popular
Altair	66	135	320	3304	15144	σ < -2	Popular
Vega-Lite	95	74	208	1592	13702	σ < -2	Popular
mpld3	38	92	319	1757	10207	σ < -4	Unpopular
ipyleaflet	47	32	147	589	6929	σ < -4	Unpopular
qqgripd	17	75	144	1527	5997	σ < -4	Unpopular
gmaps	14	18	97	485	3075	σ < -4	Unpopular
r2d3	5	38	30	316	1826	σ < -4	Unpopular

Figure 2.10: A popularity comparison between some of the most used libraries in Notebooks [28].

platforms[18] [19] have adopted either through direct integration or widgets[20]. Other very well established technologies like ggplot [21] or Matplotlib [22] might not have as many advanced features as the aforementioned but have gained the trust of their user base through their fundamental features.

2.3.3.2 Vega

Vega and most recently Vega-Lite [42] are high-level grammars that enable rapid specification of interactive data visualizations, using traditional graphical grammar characteristics such as algebra composition and visual encoding, while combining these with a new approach to grammar interactions. Vega-Lite combines a traditional grammar of graphics, providing visual encoding rules and a composition algebra for layered and multi-view displays, with a novel grammar of interaction. Vega sets to enable rapid ways to specify interactive visualization and to do some with a simple and concise systematic enumeration while pushing towards the exploration of design variation. Vega is capable of rendering bar charts, line and area charts, circular charts, dot and scatter plots, distributions, geometric graphs, tree diagrams, network diagrams, while including custom visual design tools and several techniques [51].

Either by implementing ways to visualize airport connections in the U.S. (Figure 2.11), depicting character appearances in a novel (Figure 2.12) or PI calculations (Figure 2.13), Vega provides uncomplicated and lightweight solutions. On the other hand, notebooks' versatility allows for tools like these to be used with relative ease by research scientists.

2.3.3.3 D3

D3 is a JavaScript library for manipulating data based documents, allowing to bind arbitrary data to a Document Object Model, and applying transformations to the document [11]. It reduces overhead computation allowing greater graphical complexity at high frame rates, transition sequencing through events and providing a wide array of drawing options. Being a web oriented

[18] https://jupyter.org/about
[19] https://beta.observablehq.com/
[20] https://pypi.org/project/ipywidgets/
[21] http://ggplot.yhathq.com/
[22] https://matplotlib.org/

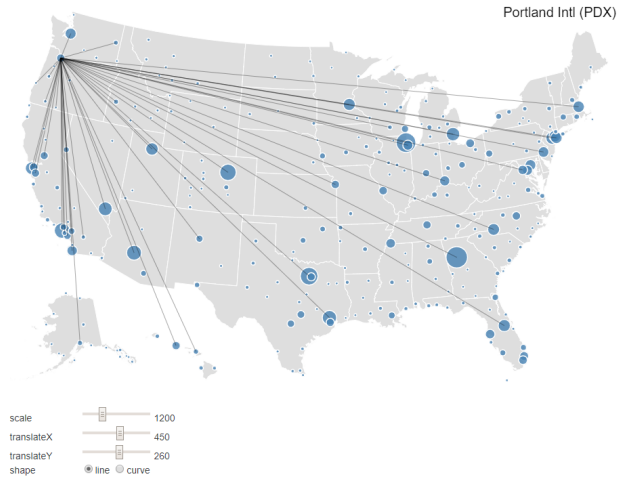

Figure 2.11: An interactive visualization of connections among major U.S. airports in 2008 [51].

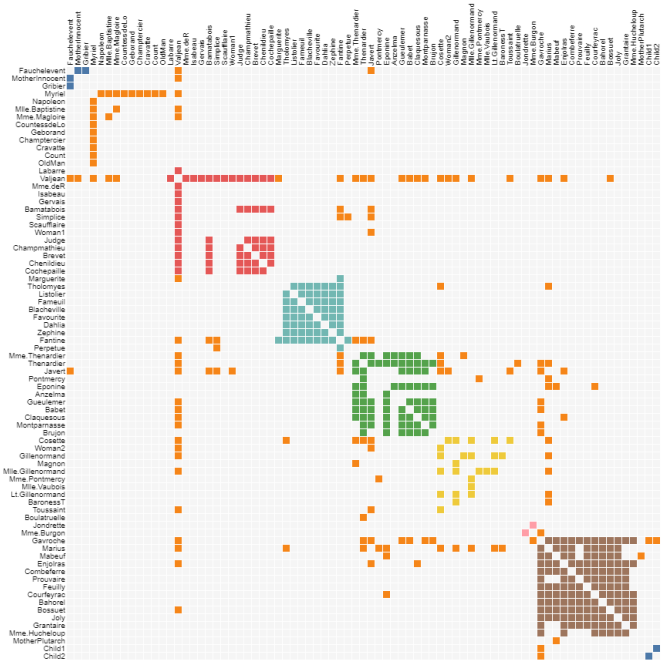

Figure 2.12: This example depicts character co-occurrences in Victor Hugo's Les Misérables [51].

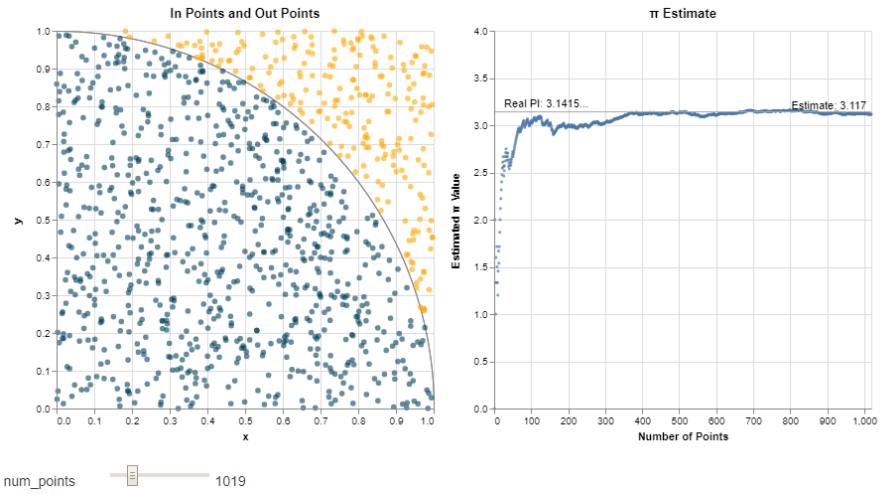

Figure 2.13: Estimation of PI by randomly sampling points and counting how many of them fall inside or outside a unit circle [51].

tool, it doesn't replace the browser's toolbox. D3 allows for instance to still use CSS3 transitions. Due to its characteristics, D3 is an excellent option for web oriented notebooks solutions being lightweight whilst very powerful.

D3 powers a very wide range of websites and is widely used in web notebook platforms such as Observable[23]. In Figure 2.14 we can see D3 being used to map the package hierarchy for a visualization tool, and in Figure 2.15 used to represent animated changes in vote shift in the U.S. elections.

2.3.3.4 Matplotlib

Matplotlib is a library written in Python limited to 2D plotting. This particular aspect may just be one of the main appeals to many of its users making plot generation, histograms, scatter plots, among others, a streamlined process achievable with a small number of lines of code. An interface similar to MATLAB's and its ability to be used within IPython notebooks contributes towards making users feel at home and surely its popularity [25].

2.3.3.5 ggplot

ggplot is an interesting example allowing for an overview on some recent trends in the field of notebooks and furthermore visualization. ggplot2 is a declarative grammar for the creation of graphics, allowing users to provide data and choose a set of variables that define the aesthetics and the desired type of visualization. ggplot on the other hand is a plotting system for Python based on ggplot2 that seeks to bring this workflow to Python users [21]. Considering ggplot2 is a R language exclusive library, the effort to port its functionalities over to Python and how quickly many users picked up on the new Python compatible library showcases a trend. Jupyter notebook's rise in

[23] https://beta.observablehq.com

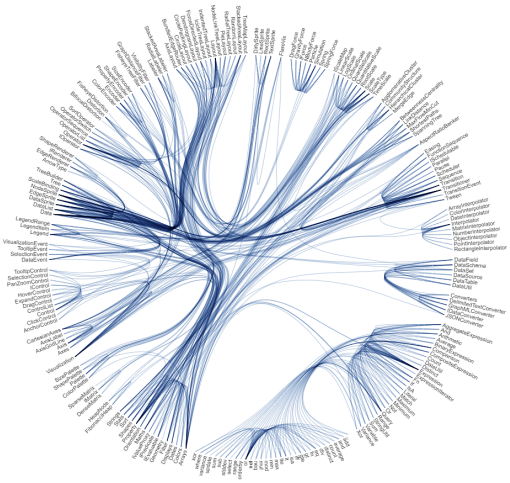

Figure 2.14: The Flare visualization toolkit package hierarchy and imports [24].

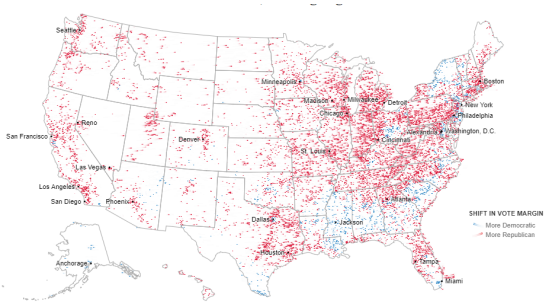

Figure 2.15: U.S. counties vote shift [11].

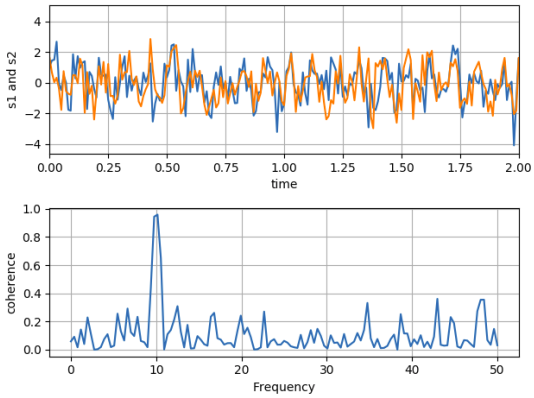

Figure 2.16: An example showing the streamlined nature of matplotlib's visualizations [25].

popularity sparked the python community into making efforts towards converting functionalities of other incompatible libraries. This happens also in the opposite direction, with python only libraries being converted to other languages. However the influx is not equivalent, bringing forward a clear trend where Jupyter Notebook's popularity alongside the popularity of Python as a programming language make it a very preferable option when considering continuous support [28].

2.3.3.6 nbviewer

nbviewer is a web application focused on rendering notebooks as static HTML web pages. This aims at providing users with a stable representation of the notebook that can easily be examined and shared with others. nbviewer is written in Python and JavaScript and uses nbconvert to render the notebooks. Its an open source project much like Jupyter Notebook and affiliates.

nbviewer does not execute notebooks, it only renders the inputs and outputs saved in a notebook document as a web page [24].

These tools provide notebooks with options for creating visualizations in a simplified way with a wide array of features. They promote interactivity with the data at hand and extend the core values the notebook has to offer. The incentive to look, interact and reiterate on scientific data promotes the fundamentals of Open Data and increases reproducibility.

After this evaluation it became clear that supporting a wide array of libraries and providing users with flexibility when deciding which tools to use in their work is crucial for the success of a platform, so we took this into account when proceeding to the implementation.

[24] github.com/jupyter/nbviewer/blob/master/nbviewer/templates/faq.md

Chapter 3

Methodological Approach

When building a systematic and cohesive methodology to approach the implementation of this dissertation's work, we looked to partition the work into phases that would guide this effort. Through this segmentation procedure we split the methodological approach into two main phases, requirement gathering and implementation. The first phase is performed in order to carefully compile a set of well defined requirements that should be fulfilled for this dissertation to be considered a successful effort. The second phase will consist in the implementation of the specified requirements. In this chapter we will review the methodology used in this dissertation while thoroughly examining all the carried out work.

3.1 Problem Formulation

In order to begin staging the implementation we first must look to clearly define what is the issue that raises the necessity of this endeavor. After the contextualization done in the review of the state of the art over the different aspects of open science, visualization and web notebooks we can formulate a problem with far better insight.

Research work comprises more than publishing an article, paper or some of the other traditional outcomes. In order to support the validity of research work research often adopts complementary goals such as description and sharing of data.

Jupyter with its web platform allowing users to build shareable notebooks was a big step forward for accessibility, allowing for sharing and reusing processing through the employment of interactive visualization, text descriptions and code execution.

To bring this sort of workflow to "Dendro" platform, whose functionalities are similar to a "private dropbox" oriented to scientific work groups that allow users to deposit and manage their files. Adding the ability to annotate this data, manipulate and interact with it, would help bringing the versatility needed to make Dendro a state of the art scientific data management platform.

3.2 Use Cases

To consolidate the problem formulation process we established a series of use cases as a way to translate this dissertation's goals into concrete coding objectives and to streamline the implementation of essential features. These are focused on the user interactions with the notebooks within the platform, from creating to editing, downloading and searching for notebooks. The IBM use-case template [1] was used as a guide for the use-case tables. These use cases, which are comprised by actors, pre-conditions and flow descriptions, are going to be user centered and assume that users are registered and logged in. Finally we define an actor as anyone or anything performing a behaviour within the system, a pre-condition as an action that must happen before the case runs, and basic flow the steps within an action in case nothing goes wrong [49].

Table 3.1: Use case table for the creation of a Notebook

Use Case 1	Creating a Notebook
Actor	User
Basic Flow	The user selects the option to create a new Notebook on the toolbox
Precondition	User is currently in a respective project folder

Table 3.2: Use case table for launching a Notebook

Use Case 2	Launching a created Notebook
Actor	User
Basic Flow	The user launches the selected notebook from the respective project
Precondition	A Notebook must be previously created at the current user's project

Table 3.3: Use case table for the execution of a Notebook

Use Case 3	Executing a Notebook
Actor	User
Basic Flow	The user runs the selected notebook executing its processing code
Precondition	A Notebook must have been previously created

Table 3.4: Use case table for the download of a Notebook

Use Case 4	Downloading Notebooks
Actor	User
Basic Flow	The user presses the download button at the sidebar of the preview
Precondition	User selects Notebook file

[1] https://www.ibm.com/support/knowledgecenter

Table 3.5: Use case table for the editing of metadata in a Notebook

Use Case 5	Editing Notebook metadata
Actor	User
Basic Flow	The user presses the edit button at the sidebar of the preview
Precondition	User selects notebook file

Table 3.6: Use case table for the previewing of a Notebook

Use Case 6	Previewing a Notebook
Actor	User
Basic Flow	The user previews the Notebook.
Precondition 1	The user selects a notebook file.
Precondition 2	The notebook file contains code or visualizations over data

Table 3.7: Use case table for the search of a Notebook

Use Case 7	Searching for a Notebook
Actor	User
Basic Flow	The user inputs the keywords for the desired Notebook
Precondition 1	The Notebook files have been created previously
Precondition 2	The Notebook file is in a public repository

After defining these use cases the next step was to consolidate the technological basis and understand how these newly added features would interact with Dendro, where it would be appropriate to integrate this newly created structure, and how they would fit into the established workflow.

3.3 Selecting Jupyter Notebook

Considering the previously defined use cases we can already start to see how the implementation will be divided into two components, the visualization of web notebooks outside of the notebook environment and the integration of the notebook environment. One intersecting aspect will be the selection of notebook technology to be implemented with Dendro. From the research conducted in the state of the art, furthered in a scientific paper written in the scope of this dissertation, we concluded that Jupyter Notebook is currently the best alternative for the average user, given its popularity and support, combined with broad support for powerful and high-level interactive visualizations [28]. The decision to integrate more than one notebook solution could possibly improve the appeal of the platform, for instance RStudio sees a large amount of popularity and is not inherently compatible with Jupyter. However it is quickly expanding and becoming a norm

which other notebook technologies try to follow [28]. Integrating Jupyter Notebook technologies with Dendro is the most feasible and flexible approach to this dissertation, combining flexibility for the users as seen in table 2.1 and clear implementation procedure without requiring a major change in the platform's structure.

3.4 Jupyter Notebook

To understand how this implementation will be accommodated within the established structure in Dendro we look to analyse how the two components of this implementation will integrate. In order to do this we must first understand the structure and work flow of Dendro. To understand where to correctly integrate Jupyter with Dendro we ought to focus on its main assignment, that is to process and to catalog research data. The established system revolves largely around concentrating users in different research projects that function as data hubs, which is where the interactions users have with data mostly take place. It makes sense to embed notebooks within projects, but one of the main disadvantages would be that each notebook would be isolated and associated to a project. However creating this association between the notebooks and projects will make the notebooks more closely connected to each project and more easily shared between members of this same project. This will allow an easier access to the research data by the notebook making it easier to catalogue and document this data with processing code or visualizations.

3.5 Notebook Viewer

When considering where it would be most appropriate to integrate notebook visualization, the inclusion within the projects component of Dendro seemed the most logical choice.

Similarly to the assets geared towards the creation and execution of notebooks discussed above, it made sense to consolidate all of the features in one place. Since file creation and upload features in Dendro converge around the project structure, the most reasonable approach would be to implement the ability to create and save notebooks associated to a project. In this manner users would be able to create a project, run a notebook environment where they would be able to access such project data and have all the usual notebook execution capabilities. In the already existing project view, an element was added in the navigation list where the notebook visualization is displayed. Initially this display was designed as a rendering tool only capable of rendering the notebooks created within the project. However, the idea of being able to render python notebooks was an achievable concept and thus the scope of the implementation changed in order to accommodate notebook rendering.

3.6 Iterative Implementation

One final step before proceeding with the implementation was to create an implementation plan that could accommodate direct feedback from possible users as a way to shape the develop-

ment. Providing an appropriate response to the requirements of the established user base of the platform was a major focus during this implementation. To understand how impactful the proposed solution will be, a series of interviews with researchers were planned. The intention was to find an expert user, whose experience with Jupyter Notebook was significant and could give insight on the typical workflow and expectations a platform such as Dendro should fulfill.

The purpose of these interviews, besides assessing expectations, was also to guide the implementation in a iterative way that would meet the expectations of users through a series of adjustments over time resulting from feedback gathered in these interviews. Initial interviews were intended to identify the desired features and perform adjustments. Later interviews, besides updating our expert user with the development, would also help evaluate the effectiveness and overall impact of the implementation on their initial vision and workflow.

In order to conduct these interviews it was necessary to carefully select our expert user, create an interview script and extract appropriate conclusions.

3.6.1 Expert User Selection

When selecting a candidate for this process the focus was on researchers familiarized with both Dendro and Notebook technologies. However it became evident early on that finding such a profile would be unlikely. The still limited scope of Dendro in comparison with other scientific data management platforms alongside made it harder to fit these requirements. The decision was made to prioritize experience with Notebooks over familiarity with Dendro. Search began within research teams in close proximity with this dissertation's work group, collaborators of INESC TEC and fellow colleagues. This is where we first came across a researcher currently using Jupyter notebooks for research work comprising large satellite data analysis. The search for other subjects did continue for a period of time, but due to the difficulty in matching the defined criteria, an interview plan was setup and contact with our already established expert user was established.

3.6.2 Interview Structure

The structure under which these interviews would occur was organized as a three phase process. First, a series of emails were sent to establish the level of familiarity of the subjects with Dendro's workflow and Jupyter Notebooks. Followed by a personal interview in order to better interpret the needs and expectations of users of these technologies. And lastly, a final meeting in order to obtain some evaluation metrics on how the obtained solution would affect the established workflow, possible improvements and the level of satisfaction with the implementation.

3.7 Implementation Strategy

After gathering and considering these requirements we could have a very rough view on how the implementation would pan out. It quickly became clear that the work would be focused on the implementation of two key components.

These would provide the basic structures for the integration with Dendro. At its core we looked to implement a structure that would allow users the ability to execute instances of Jupyter Notebook capable of processing, managing data and creating visualizations or annotations within each repository. And to complement this we also have to add another structure capable of displaying notebooks annotated with visualizations ideally through commonly used libraries like nbviewer[2].

The implementation strategy was to begin the work with the integration of Jupyter Notebooks followed by the implementation of a structure to render notebooks. In this section we will go into detail over the two main implementation steps that would allow the fulfilment of the projected use cases — integration of Jupyter Notebooks and ability to render notebooks in the "Dendro platform". We will also discuss how the interviews with our expert user influenced this implementation in both key aspects.

3.8 Expert User Interview

In this section, and before getting into the description of the implementation, we will explore in which way the interview with the chosen subject influenced the implementation through an analysis of the interview itself and the outcomes taken from this effort.

3.8.1 Interview with Expert User

After making contact through initial emails it was settled that the selected researcher was using Jupyter Notebook to capture process of handling large chunks of data, attempting to enhance reproducibility. In order to conduct the interview in an efficient manner and extract useful deductions towards the implementation, it was very important to define clear topics of discussion and objectives.

Therefore, according to the obtained knowledge through the initial exchange, discussion guidelines for this interview were centered around three main concerns. The routine workflow, associated systematic problems and personal suggestions for improvements by the interviewee. Notebooks were being used as a gateway to operate over data while preserving the methodology for re-utilization. Since most users will look to take advantage of the most basic features of Jupyter Notebooks, this can be seen as a very common usage. When inquiring about the usage of notebook's visualization potential the interviewee mentioned only the annotation of notebooks with simple visualizations provided by the nbviewer library.

Considering these use cases, the researcher was then asked about the difficulties and limitations within their workflow. Issues became apparent when analysing the data structure with which this researcher was working. Firstly the notebook was being used only to interact with data and storing the handling process and had no interactions with the Dendro file structure. This led to a tedious process which required extensive lookup in an vast repository with no control over its file structure. Some performance issues were also reported, in part due to the dimensions of the manipulated

[2]https://github.com/jupyter/nbviewer

files, the notebook provider's platform limitations and the fact that segmentation of the files was not possible.

Finally the way the current platform was set up didn't allow the researcher to share her work in a way that could make possible for it to be expanded upon by other researchers, which was in large part because it wasn't aimed at reproducibility from the start. Due to concerns with privacy and the integrity of the notebooks and with no current solution, aspects of reproducibility were greatly disregarded.

Lastly the conversation went over some suggestions on how to improve some aspects of the designed implementation with a focus on employing visualization. These suggestions in combination with an analysis of the conversation lead to a series of deductions on how to better adapt the work in development as expected.

3.8.2 Extracted Deductions

The conclusions taken from this interview revolved around three main subjects: visualization requirements, performance evaluation and Jupyter use cases.

Concerning visualization, it was emphasized that methods regarding its use were field specific and could range from crucial to redundant. Users in some fields may find it strictly necessary to accompany their annotations and work with visualizations ranging from the simple to the very complex, while other users may not find it necessary at all. In the particular case of the interviewed researcher, visualizations were used to explore data but visualization support from libraries such as D3.js[3] weren't strictly necessary. What we took from this was that a convoluted solution wouldn't be ideal. While having the option for deploying visualization libraries in a modular fashion would be ideal, it should be considered that a big portion of users might not need these features by default.

Regarding performance, this researcher in particular considered it to be a big case for concern. While concentrating notebook services in online platforms it is important to think of performance as key since daily users of the platform will have to deal with these aspects for extended periods of time. This subject in particular has faced issues with execution times in notebooks and stuttering issues. The visualization previews being used at the time were causing the researcher to loose sometimes up to hours trying to work around these issues. Having already experimented with different web browsers, libraries and methods, the researcher felt stranded due to the immutable structure of the current platform.

Ultimately, in this particular case, Jupyter was mostly being used as a way to process data for analysis and as a tool for the research document processes for personal use, often not having the detail required for others to interpret them independently. As previously mentioned the use cases for the implementation will be varied and using the notebook as just a showcase for scientific work won't be enough to fulfill the needs of the various users. It will be important to also bridge the gap between the notebook as a tool for scientific research while keeping the efficiency of its all-around functionalities.

[3] https://d3js.org/

3.9 Jupyter Notebook Integration

When starting the implementation of this notebook solution one of the first steps was to study where the integration of a notebook into "Dendro platform" would best be suited. When analysing the structure of the platform it became clear that centering features around the project structure would be the most elegant solution. Furthermore, being aware of the typical workflow is also indispensable, as the interaction users maintain with the platform mustn't suffer significant changes with the inclusion of a notebook service.

The integration of Jupyter Notebooks in this solution can be seen as a three step process building up to a fully implemented notebook service. Firstly, the usage of a docker image used to launch the notebook to be distributed by the users through a reverse proxy structure. Then, adapting Dendro to work as a reverse proxy between the user and the running Jupyter images. And finally, a synchronization process keeping consistency between the project file structure in Dendro and the notebook's file structure.

3.9.1 Jupyter Notebook Docker Structure

The first step was to chose a Jupyter Notebook image that would suit our needs. We looked to integrate the most stable and continuously supported image, so the chosen docker image was the popular and frequently updated scipy-notebook [4].

In order to understand the implementation we have to have a basic understanding of docker [16] and also understand the logic behind docker deployment within Dendro.

Docker enables the separation of applications from the core infrastructure of Dendro. This way software can be packaged and run in a isolated environment, and this environment is called a container. This isolation allows many containers to be run simultaneously on a given host. These lightweight containers run directly within the host machine's kernel. Each of the running notebooks will be a container of the scipy-notebook running in its own individual instance. These images can be started, stopped, move or deleted [15].

Before proceeding, it is imperative to make some distinctions between some core elements in this work's infrastructure. Images, containers, networks, volumes and orchestras are going to be discussed throughout this section and their definition should be explored deeper.

A **Docker Image** is essentially a file composed by layers representing instructions for the image that is used to execute code in a Docker container. It is built from the instructions necessary for a complete and executable version of an application. One image can be ran by Docker multiple times creating multiple instances of the respective container [27].

A **Docker Container** distinguishes itself from the image mostly through its writable top layer. This layer is capable of storing all the modifications over existing data within the container. This layer is unique to each container and leaves the underlying image unchanged [27]. This is advantageous for our implementation since it allows for every container to be based of the same image while still having unique data state.

[4] https://hub.docker.com/r/jupyter/scipy-notebook

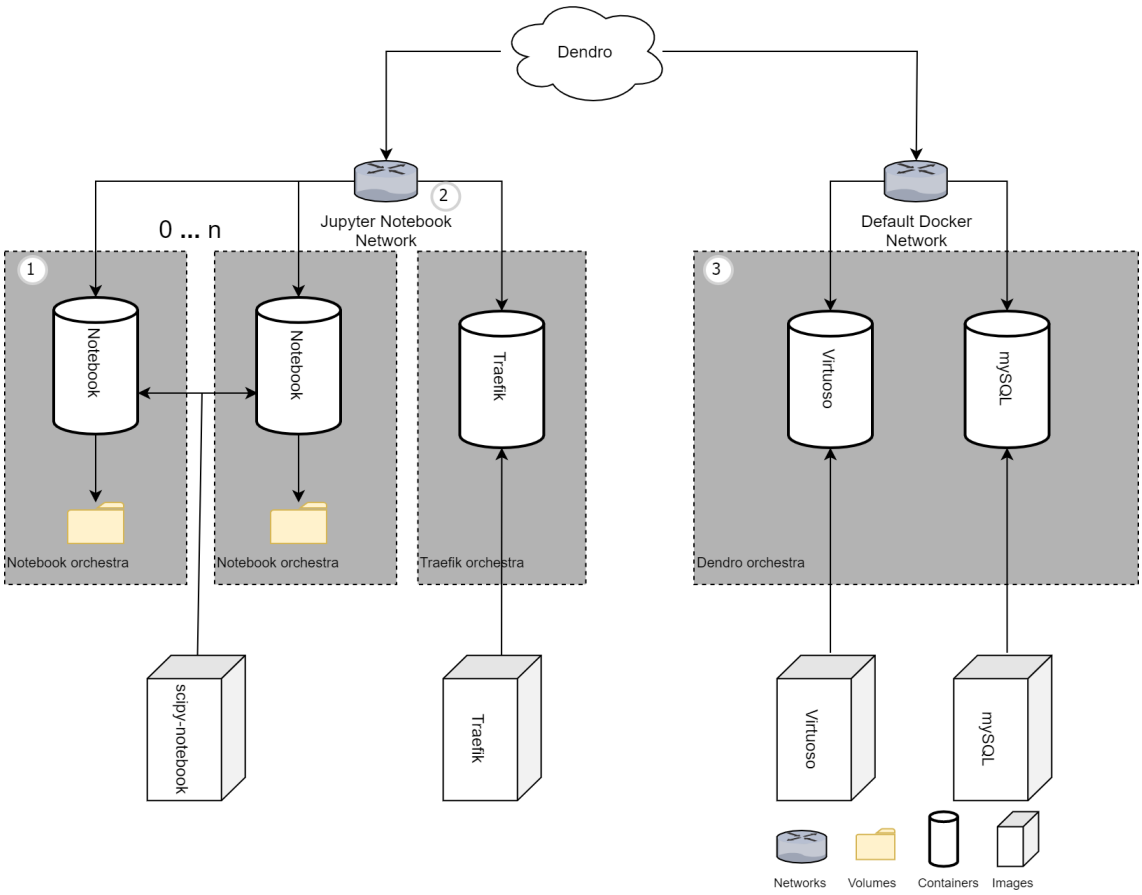

Figure 3.1: Implementation Scheme Representing Docker Structure Within the Integration

A **Docker Volume** is a directory in the Docker host's file system that is accordingly mounted directly into a container. These volumes are not controlled by the storage drive since they directly bypass it and operate at native host speeds. Multiple volumes can be mounted into one container and on the other hand can also be shared. In this implementation each notebook container is assigned one and only one unique volume [27].

A **Docker Network** is created by default in any docker host, while then automatically adding containers to it. This creates a subnet that allows containers to run within a network, and these containers can run on an isolated network (for instance for security reasons) or can run within the same network allowing the containers within the network to communicate with one another [15].

Finally an **Orchestra** can be defined as a series of micro-services, which in this case are the different containers running within each of the existing orchestrations, with a common goal. They work in a logical way to provide a service bigger than each of the individual micro-services [36].

To understand how these aspects come together in this implementation we can look at Figure 3.1, where we can see the images, containers, volumes, networks and orchestras implied in this implementation. The images "scipy-notebook" and "Traefik" were added in this implementation, while the images "Virtuoso" and "mySQL" are representing some of the already existing images in Dendro. The gray blocks represent the orchestras.

Firstly we should consider the existing infrastructure. In Figure 3.1 we see how containers are created based on the images and are orchestrated into the "Dendro Orchestra". Since no network was specified in the respective docker-compose file all of these containers are added to the default network, this can be seen in **3**.

To implement the notebook service a new separate infrastructure was implemented. Since Dendro is responsible for orchestrating both of these structures, the reason why separated orchestras and networks were created was based on two aspects. These infrastructures were fundamentally providing a different service. We look to orchestrate the notebook services in a way that allows to start, stop, delete and deliver distinct notebooks amongst the users. There is also a necessity created by the routing requisites of the implementation for the different notebooks and the routing entity to communicate with each other. This led to the creating of a separate network to accommodate the notebooks.

Referring back to Figure 3.1, in **2** we identify the newly created notebook network, any container created based on the "scipy-notebook" image is added to this network alongside the proxy micro-service "Traefik". This will allow for these containers to communicate with each other, which in this case means the proxy service will know about the existence of any running Jupyter notebook container and will therefore be findable by the proxy.

In **1** we can see each of the notebook orchestras, linked to the notebook network and respectively containing a volume for the data within the notebook. These notebook orchestras will only be launched by Dendro if a notebook is created or activated, and can go up to any number of instances. The volume, mounted on the host's file system, will contain a data folder where the corresponding data stored in the notebook container will be maintained.

Before proceeding with an overview of the behavior of the proxy micro-service there is one final aspect, within this infrastructure, we must analyse. In Figure 3.2 we have a scheme for the naming structure of the containers and volumes within the system.

Each container is assigned a randomly generated ID upon creation, the corresponding volume on the host system is then mounted within a folder called "jupyter-notebooks". This folder contains various volume folders that will share their name with the ID of the respective container as represented in Figure 3.2. These IDs are stored in Dendro at the time of creation for each notebook, meaning each notebook shares and identical volume name, container name and object name. This will be a fundamental aspect of the synchronization and redirection aspects of the notebook infrastructure.

3.9.2 Reverse Proxy Implementation

The need to correctly redirect the users requests regarding notebooks to the correct notebook container led to the implementation of a system that allows Dendro to operate mostly as a proxy for these requests. To make sure every user was correctly linked to the right container and to maintain access permissions across the platform, Dendro is required to play the role of a mediator of all requests. Therefore a reverse proxy architecture was implemented. These sort of architectures are

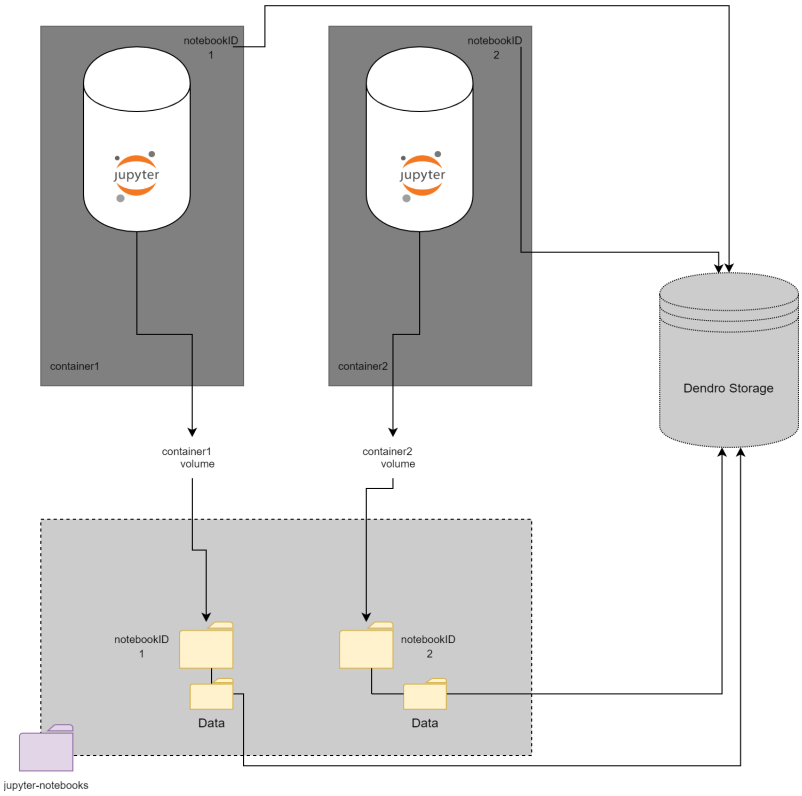

Figure 3.2: Container and Volume naming structure

usually utilized in order to achieve high availability and distribution and because of this it seemed the ideal solution for this issue.

To accomplish this, the networking service Traefik was utilized in order to simplify some of the complexity associated with creating this sort of networks. Instead of having to configure each route connecting to different paths or sub domains for each container, Traefik is used to automatically generate the routes for the containers connecting them in a simplified manner [46].

As previously discussed both Traefik and the notebook containers share the same network, which is the first requisite for the normal behaviour of Traefik. Secondly, the names of the containers are consistent as seen in Figure 3.2, which means we know internally the value of each notebook's ID, that furthermore coincides with the container name satisfying Traefik's second requisite. Being able to communicate with each of the containers, by sharing their network, and correctly addressing each notebook by its container ID, we have assembled the conditions for Traefik to correctly generate routes for the requests. Now to explore the architecture based on these features we can look at Figure 3.3.

Figure 3.3 at its core represents a request and response scenario for this new architecture. A request is made by the user regarding any existing space within the notebook. The location of the resource within the notebook is represented in the URL in gray, while the representative GUID (unique designation for each container) is represented in red. When receiving this request Dendro internally modifies it as seen in **1**, the address is modified in order to be processed by Traefik, and

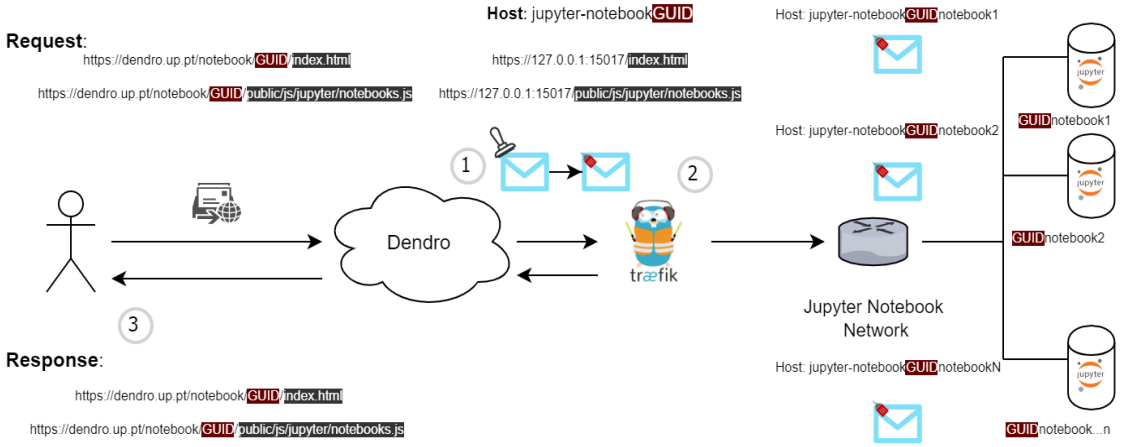

Figure 3.3: Reverse Proxy Diagram Structure

the "Host" is altered in accordance to the GUID for the target notebook. With these two steps Dendro "stamps" the request with the respective "Host" information changing it from the original "Host" of the request (that would be Dendro) to the "Host" GUID value corresponding to the target container running in the server. This modifications allow for the request to be redirected towards Traefik with this information, which in its turn will find the desired container based on the newly modified and respective "Host".

Furthermore, we can see in **2** the target resource in gray is not altered by this process. Traefik will parse the request to the specified container, without changing the URL allowing to address the correct resource.

From the perspective of the user, however, this process is seamless as seen in **3**: the URL remains unchanged and the user is unaware of the proxy process.

This allows for the same container (Jupyter notebook) to be accessed by any user that attempts to connect to it correctly. This means that any user that has permission to access the container (either by being a collaborator of the project, its creator or in the case of a public notebook), can have simultaneous access to the same instance of the executing container.

3.9.3 Synchronization and Data Management

With users using the notebook service to successfully access running Jupyter Notebook containers there is one fundamental aspect of the notebook service for consideration. Managing the data within the notebook, ensuring its availability to all users and maintaining the integrity within the different file systems. Each container encompasses its own file system, different from the one in storage, therefore this structure must be replicated in order to share the same virtues as other data objects in Dendro such as the ability to add metadata.

Users should be able to start a notebook and modify its file structure either by creating files and folders or uploading their own. The opposite should also be valid, allowing for existing notebooks' data to be persistent over time and consistent with the data existing in the repository.

There are three main systems in place in order to bring this structure together. **Notebook Creation** which is the process evoked when a user creates a new Jupyter notebook within the platform. **Notebook Restore** can be evoked by an user attempting to access a running notebook container or trying to launch an existing one — it can be most easily summarized as an attempt to access a notebook that has been previously created. Finally the last system is the **Notebook Monitor Job**, a periodic process that looks for changes within the existing notebook volumes and is responsible for updating the respective notebook objects in storage with the incoming data.

When **creating** a notebook, the Jupyter Notebook's volume (its internal file system) is defined by a parameter set on container generation, which will be the root location for any file or folder created within that notebook. Each notebook behaves much like a folder containing all of the data belonging to the notebook, including `ipynb` files themselves. The concept of notebook within the platform won't represent the traditional notebook `ipynb` executable file but will instead work much as an object containing all of the notebook's structure files and data. Making each notebook a fully independent and isolated entity that can be shared and exported.

When **restoring** a notebook there is a series of checks that must be performed, which are centered around two main components necessary for the correct execution of the notebook service. These are the current state of the docker container (running or stopped), and the existence of the volume folder in the host machine. This process will then be in many ways similar to the creation of a notebook for the first time, and we can overview this in full effect in Figure 3.4.

Before analyzing Figure 3.4 in depth, the decision states for which an user can restore a notebook should be clarified. There can be four possible scenarios related to the container and volume state:

- The most straightforward scenario would be a running notebook container and an existing volume within the host machine, this means a Jupyter notebook is active and the user accessing it just needs to be correctly redirected.
- The second scenario would be a stopped notebook container while the data volume exists in the host machine folder, this can be caused by a crash in the container, for instance, and the container is then restarted.
- Another scenario would be an active notebook container but no data volume in the host machine, this could be caused by an error or a platform crash when restoring the Jupyter notebook object data, the volume should be restored and the container restarted.
- Finally, the last scenario would be the default scenario for an existing notebook in the platform, the container is stopped and the volume data is not mounted on the host machine.

In Figure 3.4 the processing flow of the create and restore operations can be seen. After the user selects the option to create a new notebook, the current folder within the project structure will become this notebook's parent folder and the process of starting the orchestra will begin. Concurrently, a volume is mounted on the local system and the notebook's ID and details are saved

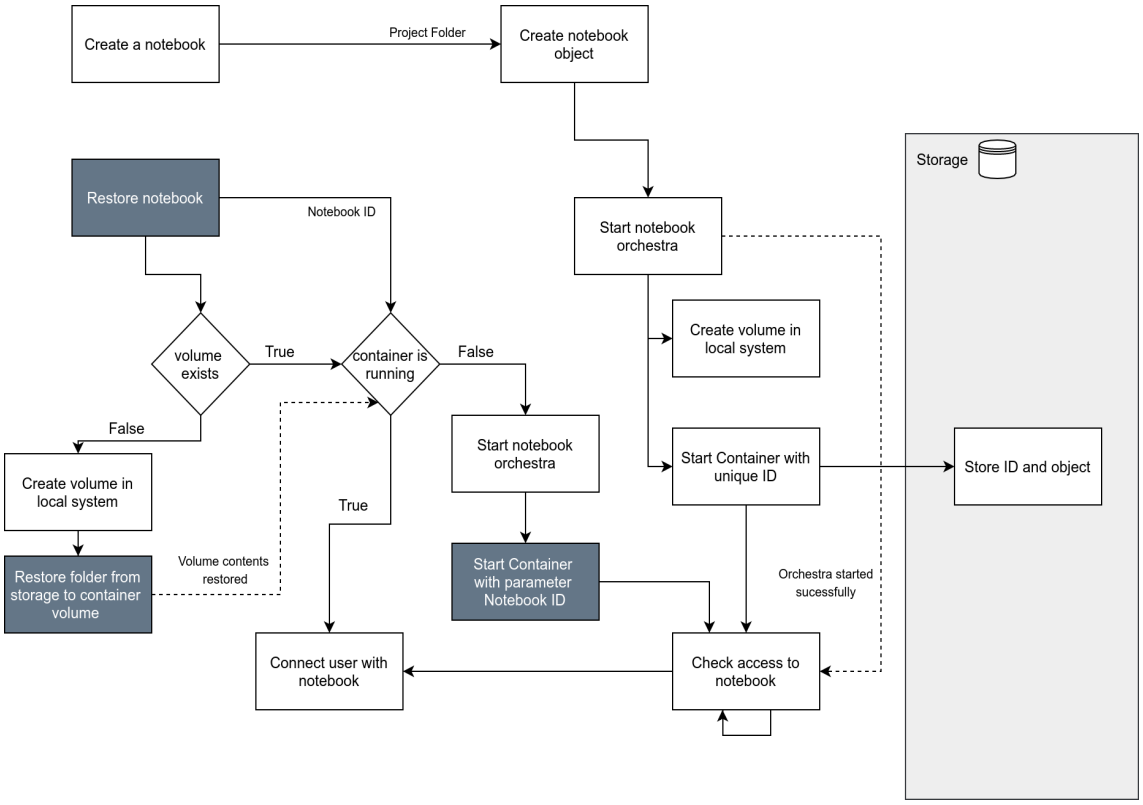

Figure 3.4: Flow Chart of Notebook Creation and Restoration Systems

in Dendro storage; once these operations are complete, Dendro will access if the container is ready and will redirect the user to the Jupyter Notebook splash page. When choosing to activate/restore an existing notebook, the process will be somewhat similar, with the addition of the corresponding checks for the status of the container and volume. After these checks, the process will be merged at the respective phase as seen in Figure 3.4. The processes in the gray boxes are exclusive to the restore functions.

The final aspect of this system is the monitoring job process. Jobs are a series of processes that are executed by the server running some process independent from the user. Using *cron* syntax to define the interval of time to which the "Notebook Monitor Job" would run, which is currently set to run every five minutes, this job is then responsible for comparing the value of the *last modification* within each of the running notebooks with the *last modification* value in the repository for the respective notebook. It will then proceed to upload the data from the notebook to the repository accordingly.

In Figure 3.5 we notice the workflow of the job, launching on Dendro startup, which asynchronously performs the checks for the modification values and then performs the comparison for each of the notebooks. As, previously described, if the modifications in the volume are more recent than the ones in the repository the data is uploaded. However if the modifications in the volume are not more recent than the ones stored in Dendro this means that the modification values are the same and the current content is already up to date, which means we are not free to shut

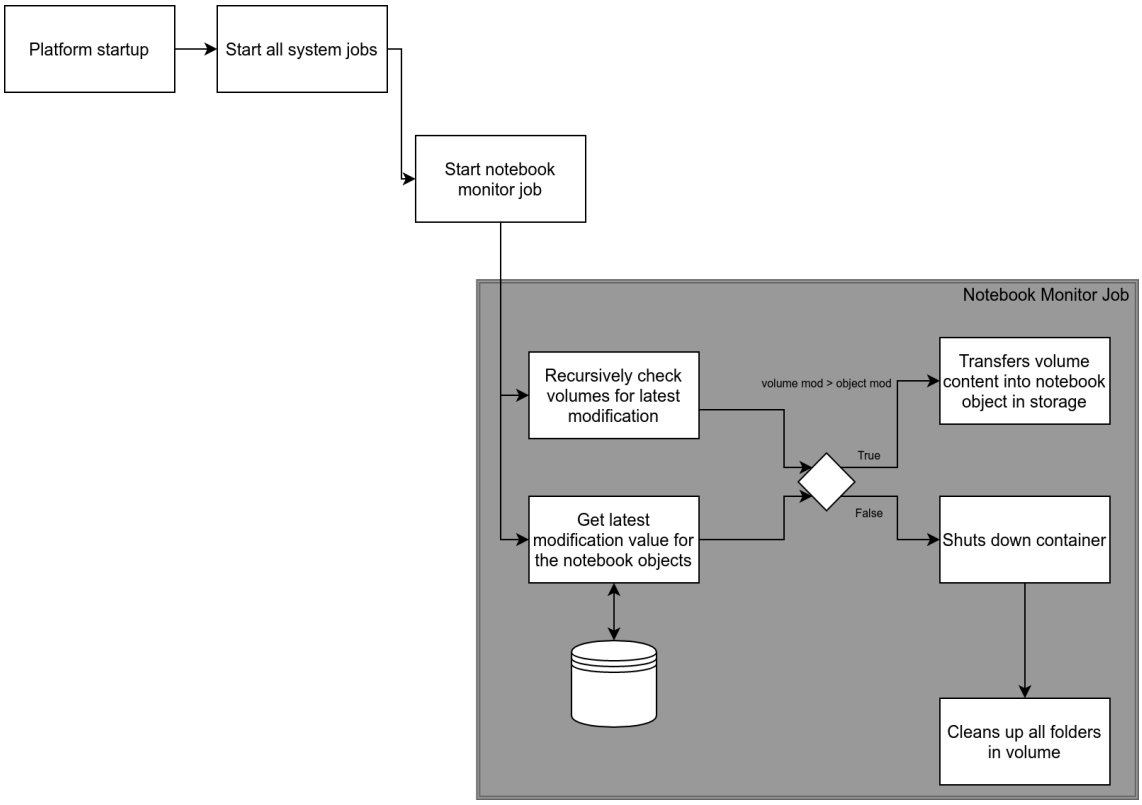

Figure 3.5: Notebook Monitor Job schema

the container down and clean up the volume folder as seen in Figure 3.5. This means that each notebook has a current timeout value of five minutes and after that the container will be shutdown and will have to be restarted. This guarantees that the server is not overloaded with an excessive number of notebook containers running simultaneously and data is uploaded to the repository at reasonable rate. These values can modified and adjusted accordingly.

3.10 Jupyter Notebook Viewer

In order to convey more versatility to this web notebook service within the platform, it is fundamental to have the ability to quickly view notebooks. These previews should be available to the notebook creators, the work group and or publicly if the owner decides so. The central focus when deciding the aspects of this visualization method was which technology to deploy. The rendering of the notebooks on these pages seemed only logical to be done with *nbviewer*[5], as it is one of the preferred technologies for fast notebook rendering.

As seen in our structure model, the most logical way to integrate notebook previewing should be within the context of the project allowing for users to preview a selected a notebook by exploring a notebook folder.

[5]https://github.com/jupyter/nbviewer

Even though previously studied in the state of the art analysis, it is once again interesting to scrutinize the way nbviewer was implemented in this solution since it uses a different logic from the original nbviewer. The current version of nbviewer[6] provides rendering functionality to a series of notebook technologies, when provided a repository link containing a notebook. The local running version of nbviewer provided by Jupyter adds more features and higher versatility since its functionalities can easily be extended. The implemented version is based on a client side rendering version[7] of nbviewer. The reason why the work was based on this implementation was because of its lightweight nature and possibility to render a notebook without any running instance of Jupyter Notebook.

This action was implemented in order to increase accessibility and reduce time spent in order to load different notebooks. The rendering process of the notebooks is very fast, since it is fully written in JavaScript while deploying marked.js[8] for markdown rendering and Prism.js[9] for syntax highlighting.

Finally, when implementing this rendering feature, the initial scope was to create previews for notebooks created within the project, however this feature also extends as a standalone notebook visualizer. This will allow researchers within the platform to more easily explore other notebooks and bring benefits for users using Dendro as a tool in their daily workflow, significantly increasing this implementation's appeal to a diverse set of Jupyter notebook users.

3.11 Difficulties and Obstacles

When implementing the features in this dissertation, some aspects proved to be significant obstacles. The first and most prominent was the concurrent computation required to operate a large number of notebook containers within Dendro, which was mitigated by usage of docker images.

However, this obstacle returned when discussing the synchronization of the notebook file systems and the data in the repository. The initial planned structure involved a number of watchers over each notebook file system responsible for notifying Dendro of changes within the notebook, but it became apparent that such a dynamic would cause a severe overload of the database several notebooks running within the platform.

This structure ended up being abandoned and replaced by a job that itself raised some issues at times because its periodicity could cause some data to be lost, and therefore some attention was required and several parameters had to be added to the notebook structure in the database in order to guarantee that it maintains its integrity.

[6] https://nbviewer.jupyter.org/
[7] https://github.com/kokes/nbviewer.js
[8] https://github.com/markedjs/marked
[9] https://prismjs.com/

3.12 Digressions from the initial implementation

From the described use cases that set the goals for this initial implementation plan, a standout feature that did not come to fruition was the ability to search for notebooks within the platform. Since the structure chosen for the notebooks and the way they integrate with Dendro requires the notebook to be contained within a project and retain some structural integrity because of its associated data, it became complicated to implement a searching system that would be exclusive for notebooks. Users however still have the possibility to share these notebooks within the platform, either directly with their contributors on the project, or by sharing the notebook link guaranteeing that the notebook is contained within a public project. And even though a specific notebook search functionality has not been implemented, it is important to mention that due to the notebook contents synchronization with Dendro's file structure. This allows the contents of the notebook to be annotated and described having their metadata indexed by Dendro.

Chapter 4

Outcomes and Experiments

In this chapter we describe the outcomes of the implementation and the experimental process. We will showcase the core implemented features individually. We will also document the experimental process and reflect upon the obtained outcomes extracting considerations relative to the success of the implemented work.

4.1 Implementation Outcomes

Firstly we provide a description of the implemented features alongside a showcase of the interactions users can have with the dissertation work. We will match and compare the accomplished features of this work with the initially outlined aspects. So that later we can also evaluate their implementation through an experimental process.

We divide the implementation outcomes in two main components, the integration of the Jupyter Notebook service and the notebook viewer component.

4.1.1 Jupyter Notebook Integration

In this section we explore the outcomes of this implementation from the standpoint of the user. Based on our use cases, users should be able to create, launch, execute, download and share their notebooks. Since the *notebook object structure* inherits a lot of properties from existing elements in Dendro, such as *Folders*, qualities like sharing and downloading are accessible under the previously established methodologies. However, actions like *creating a notebook*, *starting or restoring a notebook* and *executing actions within the notebook* required a totally new interface and batch of methods.

Firstly we begin by looking at how users approach the **creating a notebook** action, in Figure 4.1. The user navigates to any existing project, and after creating a folder the user can select it by left clicking on it. If the folder is selected the user can select the “Create Notebook” option from the drop down menu or it can bypass this by simply left clicking on the folder and equally selecting the “Create Notebook” option. This can be done for any folder except the root folder

of the project in order to maintain the consistency of the repository between projects that contain notebooks or not. Also, a notebook object must always be created within a folder. The user interface for the described action can be seen in detail in Figure 4.1 as previously mentioned.

When a user looks to **start a previously created notebook**, the action is very similar to the one described for the creation of the notebook. However, when selecting the drop down menu or right clicking the notebook object like previously, the prompt will be different, giving users the options to "Start Notebook" instead, as shown in 4.2.

After both scenarios of creating or starting an existing notebook, the users will be redirected to **Jupyter notebook object splash page** where they have a series of controls provided by the Jupyter container, with options to upload files, create new folders, create new files, among others. In Figure 4.3 we can see an example of this, where two files were uploaded, but these files could also have been created by the user within the Jupyter interface. The `ipynb` file seen in the figure is the typically recognized Jupyter notebook file, that in this example explores the "Lorenz system of differential equations".

Furthermore, we can see how the active kernel of the notebook is capable of **executing the notebook** in real time, alongside any imported libraries to the notebook object, allowing it to render in real-time a representation of these differential equations which the user can explore and execute at will, as seen in Figure 4.4. This demonstrates the capacities the notebook has to showcase research data in real time and perform methods of data processing within the platform.

This completes the showcase of possibilities the Jupyter Notebook framework makes available to users. However with Dendro we can dig further into the platform possibilities since through the navigation tools in Dendro we can **explore the files within the notebook**, as seen in Figure 4.5 and combine the notebook file system with traditional metadata records. In Figure 4.5 we can see the existing files within the previously created notebook, and perform any of the platform established functionalities on them.

Since each notebook works as an isolated system, the possibilities for the libraries and functions each notebook can perform are limitless and allow users to customize their notebook to their needs. One of the advantages of having a centralized notebook service system serving containers to different users is that any *dependencies* or *imports* necessary for the correct performance of the notebook won't have to be replicated by other researchers interested in exploring that notebook's work. Since each notebook and its associated structure is saved and replicated (imperceptibly to the user) on notebook start. This brings one of the biggest advantages of this system by allowing the fast and simplified reproduction of methods.

4.1.2 Notebook Viewer

Finally, on the notebook viewer section of the implemented work, the goal was to integrate a way of previewing notebooks within the project structure. As it can be seen in Figure 4.6, a section within the central panel in the project explorer menu called "Notebook View" allows for the quick preview structure to generate a static version of the notebook, allowing users to understand if such notebook is of interest.

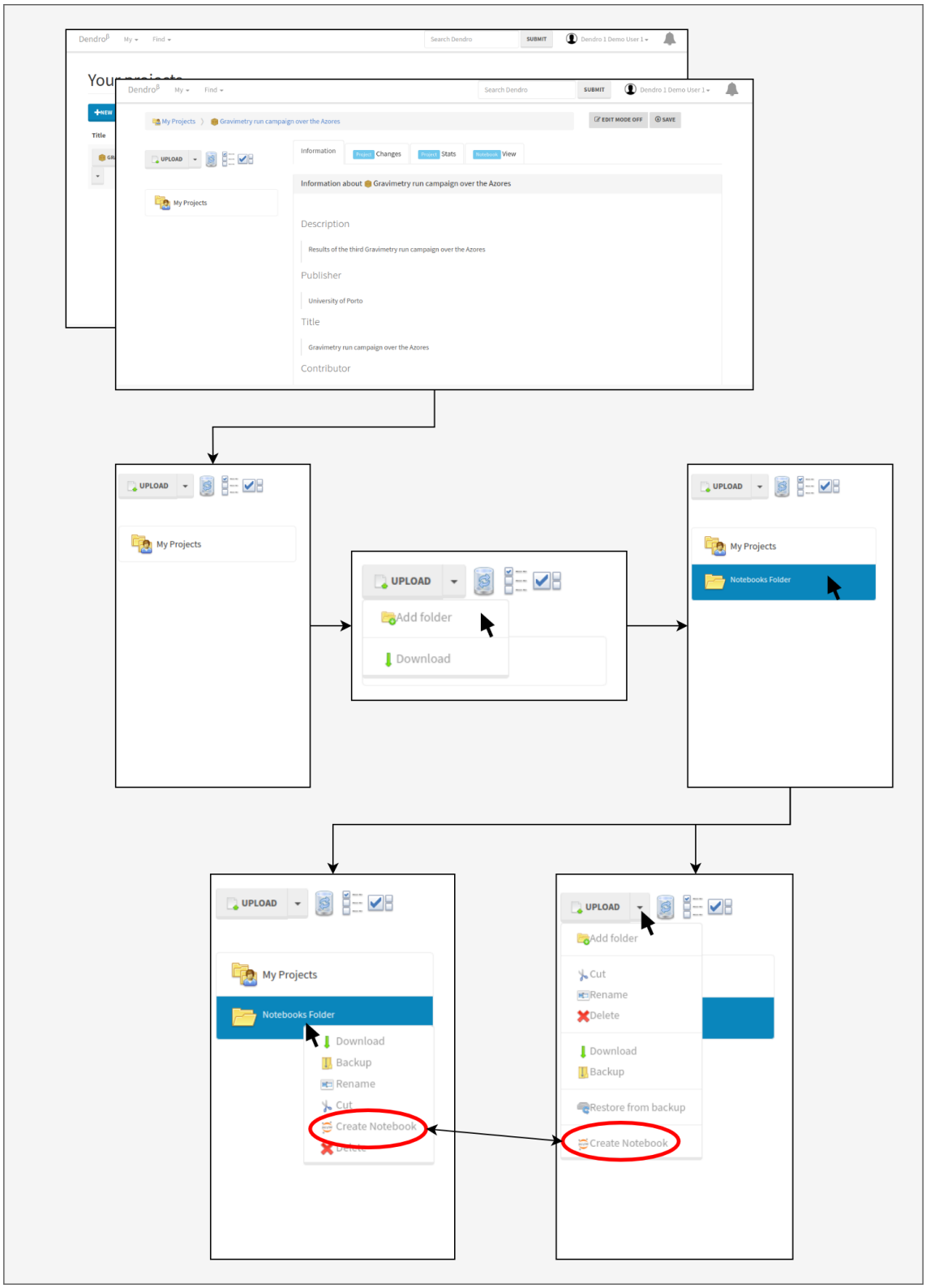

Figure 4.1: Notebook Creation Flow Chart.

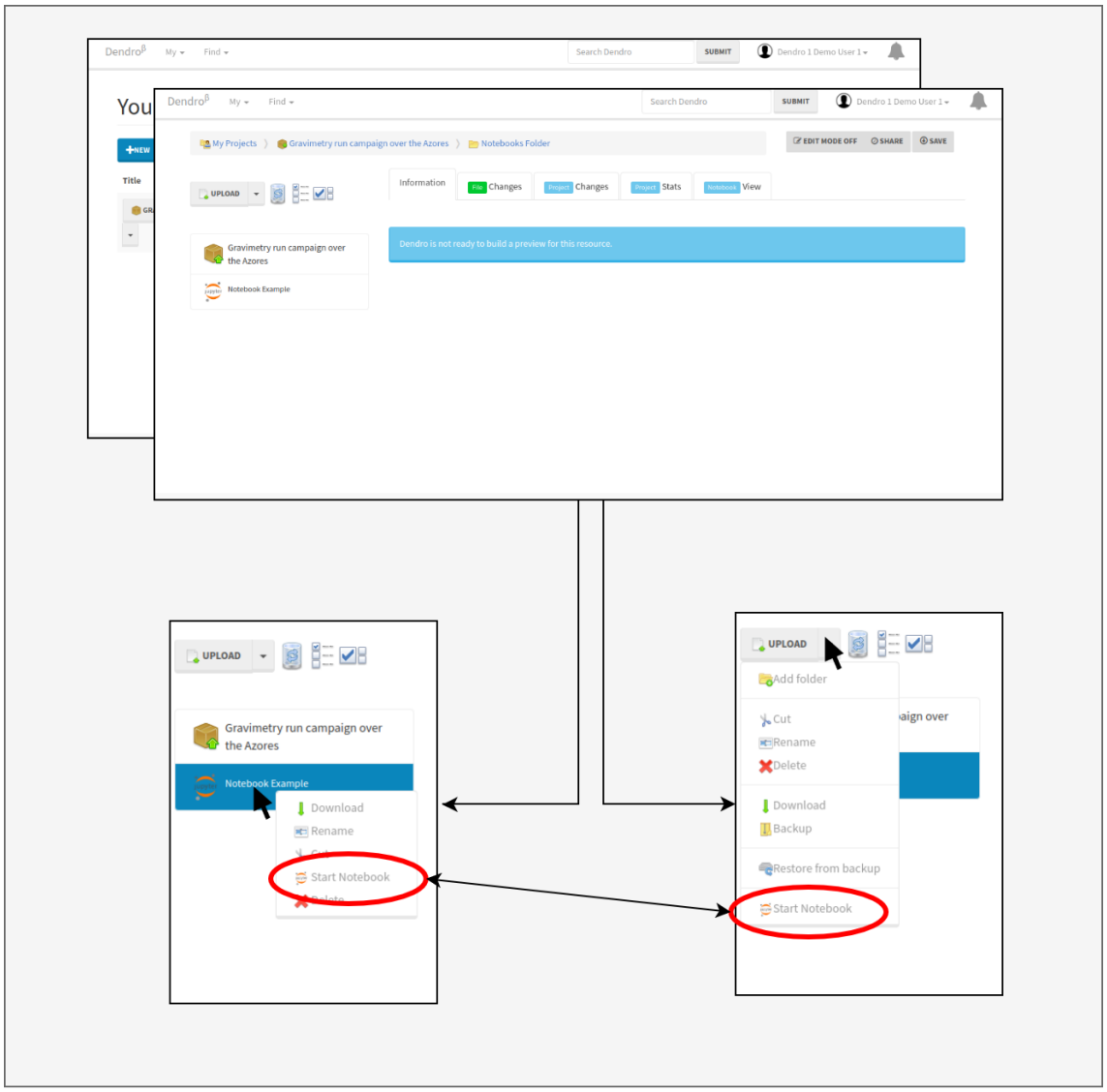

Figure 4.2: Notebook Start/Restore Flow Chart.

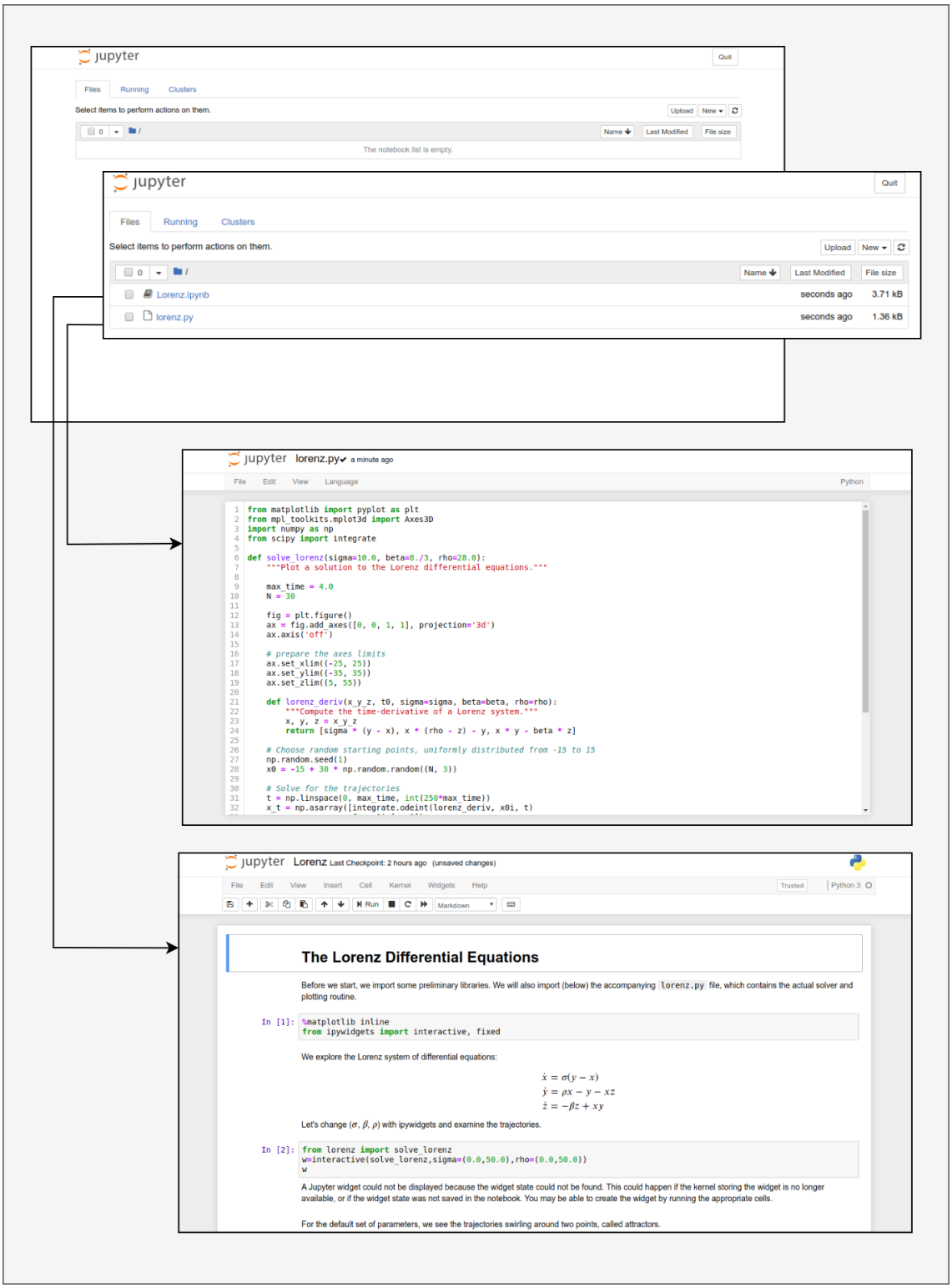

Figure 4.3: Active Notebook Interface.

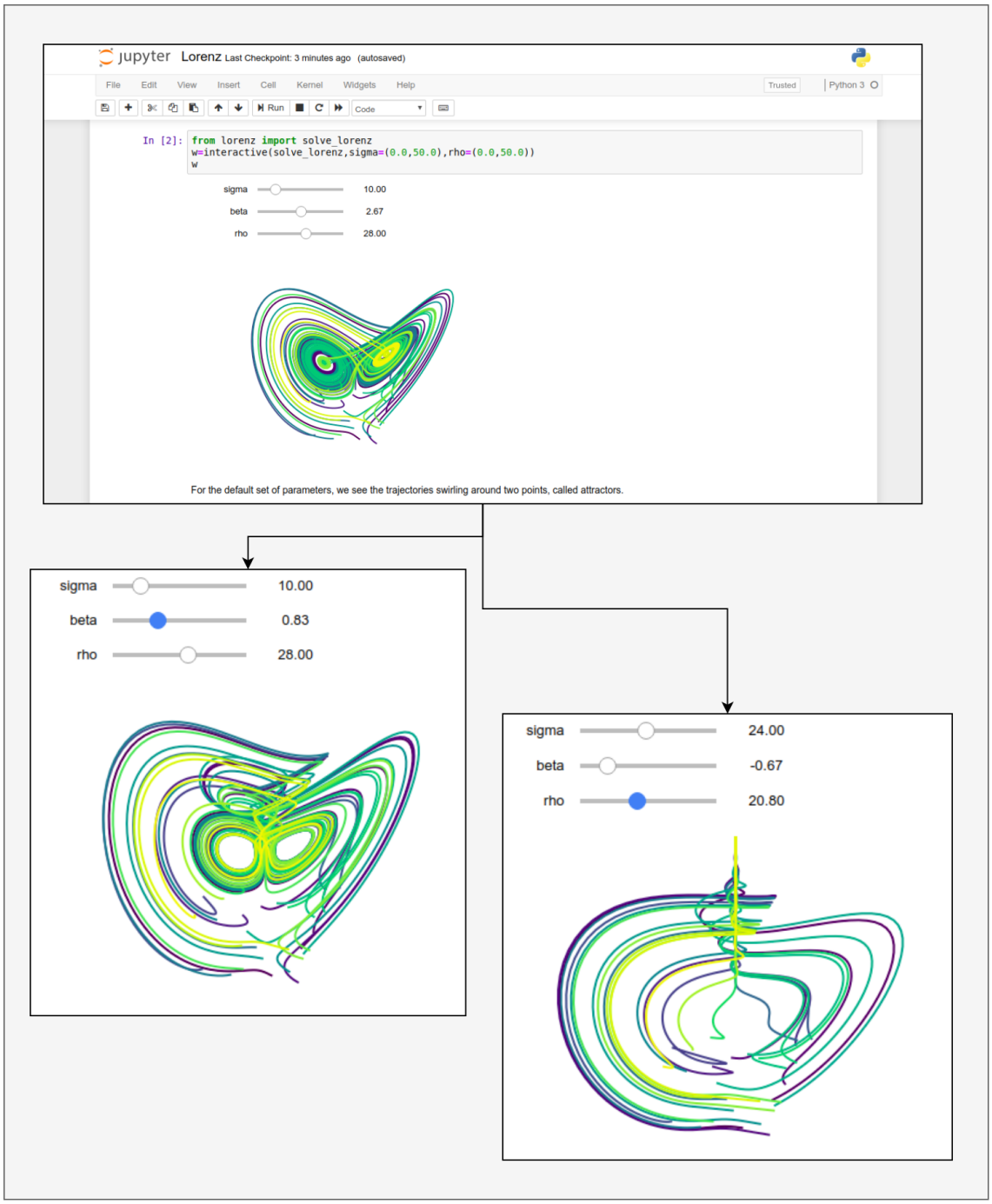

Figure 4.4: Interaction With Active Notebook Execution.

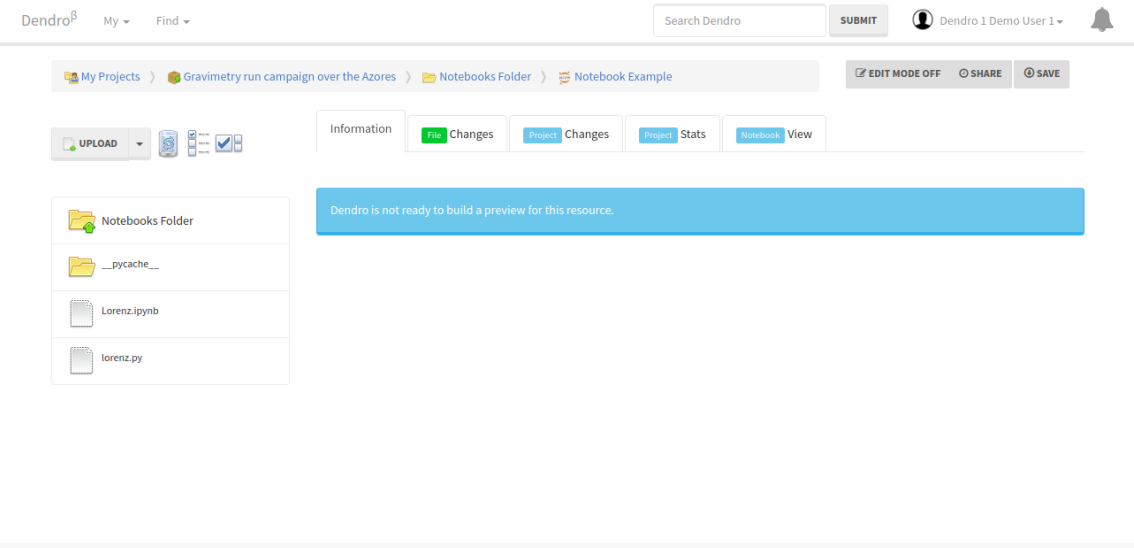

Figure 4.5: Using Dendro to Explore the Notebook File System.

This section also allows for external notebooks to be imported and rendered client-side by a *drag and drop* functionality that allows users to simply drop their `ipynb` files on the viewer section that will proceed to render the notebook. This lightweight method allows more flexibility for researchers, in case they need to manage several notebooks and need a quick way to preview them without uploading them to the platform, and an example of this can be seen in Figure 4.7

4.2 Experimental Outcomes

In this section we will be analyzing the methodology and results obtained from the experimental process targeting the developed work. In order to obtain insightful conclusions on both the usability and relevance of the implemented features we looked to perform experiments that would provide results on both the system usability and the usefulness of the implemented methods. To that end, the experimental procedure focused mostly on establishing a usability test plan that would guide participants through a series of tasks concluding in a system usability survey and an interview in order to assess the subjects' satisfaction with the work implemented. We will now analyze this procedure from initial test planning and subject selection, experiment execution and finally assessing the outcomes.

4.2.1 Usability Test Plan

This usability test plan was elaborated largely based on the proposed approach by *Usability.gov*, which served as a guideline for this section [5, 44]. *Usability.gov* is the leading resource for user experience and best practice guidelines, provided by the United States Department of Health and Human Services, this website overviews user centered design process and covers methodologies and tools in order to make digital content more useful and usable [44].

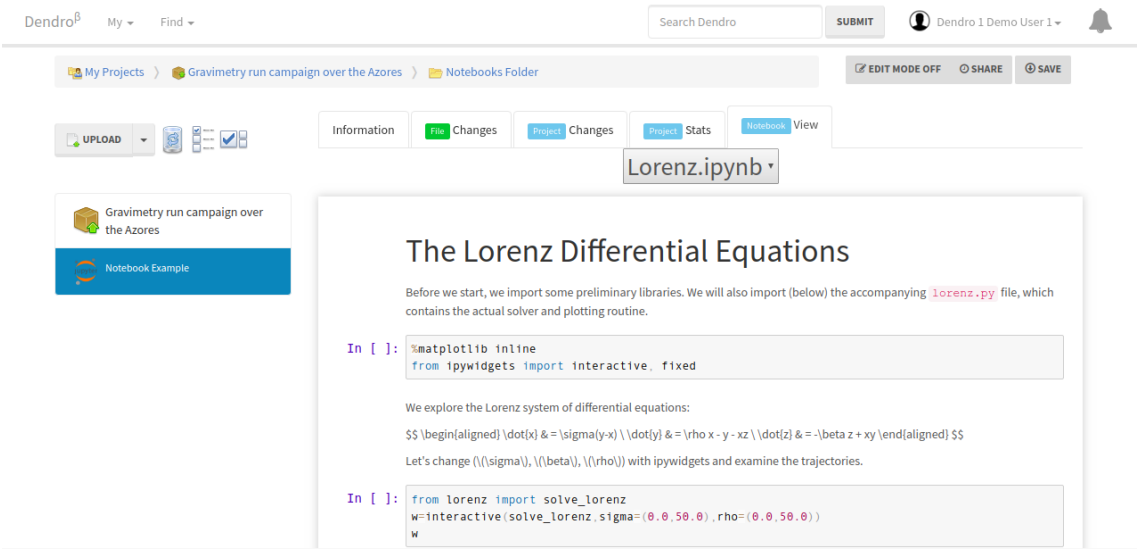

Figure 4.6: Notebook Viewer Static View.

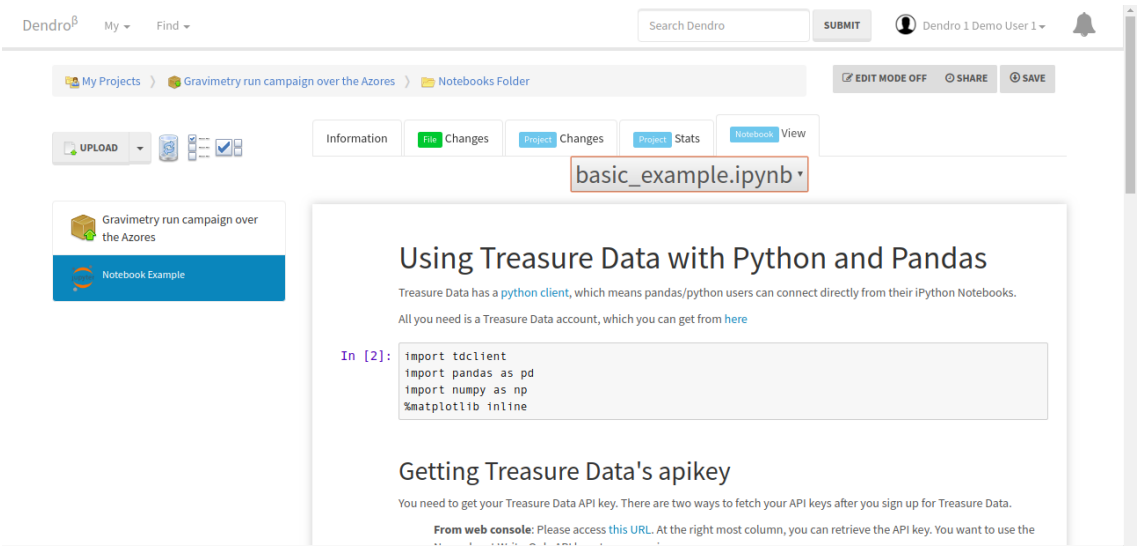

Figure 4.7: Notebook Viewer Rendering Imported Notebook View.

We carried out the usability test in order to establish a baseline of user performance, validate user performance measures and identifying potential design flaws to be addressed in order to improve the efficiency, productivity and end-user satisfaction [48].

In order to establish a baseline for user performance and satisfaction levels, a series of tasks to be performed by representative testing subjects was envisioned. These tasks would help determine design inconsistencies and usability problems within the user interface and content areas. Throughout these tasks these problems are categorized into critical and non-critical errors. Critical errors are defined as a deviation at the completion level from the targets of the scenario, where the users may or may not be aware that the task goal is incorrect or incomplete [48]. Non-critical errors are divided into three potential types of errors as categorized by [48]:

- Navigation errors – failure to locate functions, excessive keystrokes to complete a function, failure to follow recommended screen flow.
- Presentation errors – failure to locate and properly act upon desired information in screens, selection errors due to labeling ambiguities.
- Control usage problems – improper toolbar or entry field usage.

The data collected from these tasks was used to assess whether the implemented work achieves an efficient and effective user interface.

When looking to select a group of users to perform the evaluation tasks aimed at assessing usability, we looked to choose at least five participants. The reason from this number comes from the assumption that elaborate usability tests often are associated with a waste of resources. The best results come from testing with no more than five users alongside the execution of as many small testing sessions as possible [33]. To perform these testing sessions we looked to define what would be a representative end user of the platform. We characterize this user group as Dendro users whose work deals with data management and processing. Users with an interest in making their processing methods more easily accessible to contributors or research partners. Hence, the profile of a representative subject for this experiment would be a user that holds experience working with web notebooks and data management platforms, notably subjects with an interest in sharing methodologies applied in their scientific work with others. We can also consider subjects that have experience only in one of those categories, since we must account for convenience sampling, given the still somewhat limited usage of web notebooks within scientific research communities in close proximity.

The testing occured at a predefined room in which a computer running the implemented code would be available for test subjects to carry out the evaluation tasks. The subjects were assisted by a facilitator that collected the test data performing the role of data logger for the experience. Finally, subjects were asked to fill two forms: one relative to the usability features of the system and another relative to the overall satisfaction with the implemented work.

Each testing session was planned to last twenty to thirty minutes and the initial period for testing would be a time frame of two weeks within the first half of February 2020.

4.2.2 Experimental Sessions

The experimental procedure sessions took place in the Faculty of Engineering of the University of Porto (FEUP), counting with the participation of members of Information Systems Laboratory (InfoLab), a group based at the Department of Informatics Engineering at FEUP that brings together teachers and researchers in areas such as Information Management, Information Retrieval and Information Systems [20].

Eight participants were interviewed and asked to perform a total of five tasks in the most efficient and timely manner in order to later provide feedback regarding the usability and acceptability of the user interface alongside some impressions of the implemented work impact.

Participants were chosen based on their similarity to our end user group, through contacts with research partners, platforms users and collaborators. Users with no experience working with the platform wouldn't be desirable. In this manner and through convenience sampling the selected subjects were a group of associates of InfoLab or subjects in proximity with them, that are actively working within research groups and generally involved in work either with data processing, data management and computational notebooks.

Before beginning, each subject was provided with a short overview of the platform and some guidelines about the user interface or workflow necessary for the execution of the tasks but not directly related to the test, alongside a document describing the procedure which can be analyzed in Appendix C. Participants where then asked to examine a list of tasks and perform them in order using only the provided computer and an instance of the Dendro website. The subject interaction with the website was monitored by a facilitator, in this case the author of this dissertation, that was also responsible for cataloging the time taken to perform each of the tasks and monitored the session.

Based on the core features of the implementation and the formulated use cases, participants were asked to perform tasks that would cover all of these aspects of the implemented work while enveloping the common workflow of the platform. The tasks are as follows:

- Creation of a Notebook
- Upload/Creation of Notebook Files
- Executing the Notebook
- Sharing and Starting existing Notebook
- Previewing an External Notebook

These tasks contained simple instructions and should be completely explicit for the users allowing them to perform the tasks without external interference. They are detailed below.

Creation of a Notebook: The goal of this tasks is the creation of a notebook. In this task you will begin at the splash page for the website. An account has been previously created alongside different projects within the platforms, you are already logged in. Your first task is to navigate

to the view containing the listing of available projects in the platform. You must then select the default project, create a new folder and proceed to create a notebook.

Upload/Creation of Notebook Files: The goal of this task is to create a folder structure and upload files through the notebook. In this task you will begin at the Jupyter Notebook splash page of a previously created project containing a notebook folder. You will then proceed to use the notebook file system to create two distinct folders and upload one or more designated files from the computer to either of these folders. Make sure both files are uploaded on the same location: -lorenz.py -lorenz.ipynb

Executing the Notebook: The goal of this task is to exemplify the typical workflow for data processing and annotation in the implemented work. In this task you will begin at the notebook folder. In order to save time initialize the "lorenz.py" file where any code changes can be made. Launch the "lorenz.ipynb" notebook, this would be the typical processing and visualization notebook. You are now free to run the code at will, and explore the parameterization. Finally you must copy the address link for the notebook in question, save and close the notebook.

Sharing and Starting existing Notebook: The goal of this task is to emulate the sharing process of a notebook. In this task you will begin at the splash page, where you should logout of the current user. You should login with the credentials: user:demouser2 password:demouserpassword2015 This user has been previously added as a project collaborator. You will then proceed to navigate to the previously created notebook and check if the notebook is according to the one previously explored. Finally you must paste the link address in the navigation bar and assess if the URL redirection is functioning correctly.

Previewing an External Notebook: The goal of this task is to access the preview display of a notebook. In this task you will begin at the splash page for the website. An account has been previously created alongside different projects within the platforms, you are already logged in. Your first task is to navigate to the view containing the listing of available projects in the platform. You must then select the previously uploaded "lorenz.ipynb" from its local version and navigate to the notebook view tab. Then you should select a notebook file from the computer and drop it in the notebook view tab rendering its preview.

The first two tasks "Creation of a Notebook" and "Upload/Creation of Notebook Files" look to establish a foothold on the usability of the implemented service. While 'Executing the Notebook" looks to let participants familiarize with the full workflow for the creation of the most basic and typical notebook user case. "Sharing and Starting existing Notebook" evaluates the adequacy of the sharing and launching process. And finally, the "Previewing a Notebook" task looks to assess the usability of the notebook viewer feature implemented in this dissertation and the advantages it can bring to users of the platform. In the end of these tasks users would then answer a system usability test and have a small discussion about the value of the implemented features.

4.2.3 Result Analysis

Here we present a brief overview and analysis of each of the tasks and sections of the interview followed by a discussion over the extracted deductions.

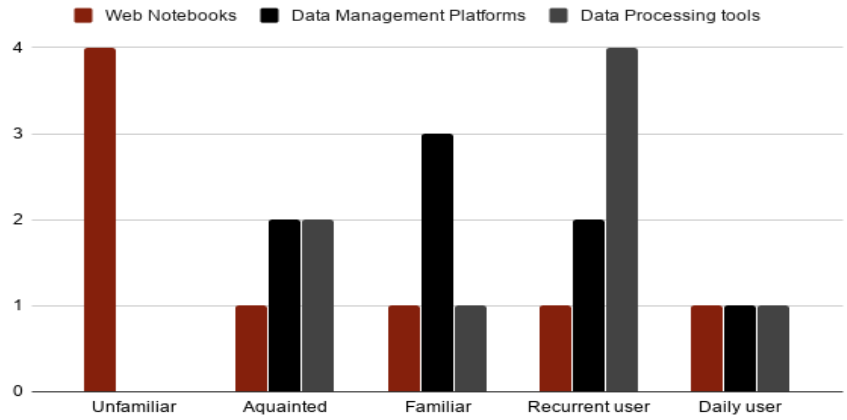

Figure 4.8: Characterization of subjects through level of experience with web notebooks, data management and processing.

4.2.3.1 Test Subjects

In order to contextualize the ability of the test subjects within the context of this dissertation the first step was to assess their level of experience with the technologies in question.

A total of eight participants were inquired. Four of these had ages comprised between 20-25, while the remaining were between 25-35 years of age. They included two female and six male participants. Each participant was asked their level of experience with web notebooks, data management platforms and data processing tools from a scale ranging from unfamiliar to daily user of the respective technology.

In Figure 4.8 we can see that from the target group of subjects we chose to interview many are familiar with data management and processing from their work in InfoLab. However, as seen in the figure, the usage of web notebooks is less common among our testers, as four users rank as "Unfamiliar", indicating that they have had no contact whatsoever with notebook technologies. This however didn't present an obstacle to our experimental method since all of the subjects were briefed on web notebooks and would interact with the implemented features in their tasks to an extent that would allow to provide some overall considerations about the implemented work. Furthermore, experience with data management platforms similar to Dendro is substantially more valuable when providing considerations of the ability of this dissertation to influence reproducibility.

4.2.3.2 Time to Complete the Tasks

The first metric used to assess the usability of the implemented functions was the time to completion of each task. In table 4.1 we can see the time each of the subjects took to perform their tasks. All tasks on average, except for the "Executing the Notebook" task, were completed in under two minutes by users that had no previous contact with the implemented work. Since tasks were self contained and always included the full exercise of actions in order to obtain the desired

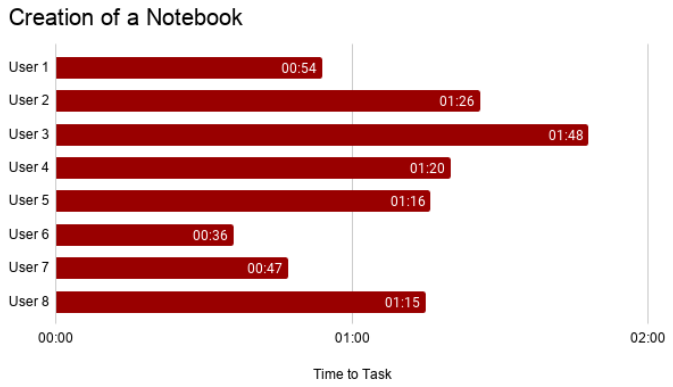

Figure 4.9: Graphical representation of time to task in "Creation of a Notebook".

goal, the critical features of the notebook service are accessible in a reasonable time. This is a good indicator that the implementation is effective in conveying the information necessary for a correct usage of the newly added aspects to the platform. It should also be noted that the task "Executing the Notebook" allowed users to freely explore the characteristics of the implementation so the time to task is not a significant indicator since different users spent more time exploring while others more quickly looked to resume the task.

Table 4.1: Time taken to complete each of the experimental tasks

	Creation of a Notebook	Upload/Creation of Notebook Files	Executing the Notebook	Sharing and Starting existing Notebook	Previewing an External Notebook
User 1	00:54.07	02:53.09	01:44.31	01:46.49	01:51.02
User 2	01:26.08	01:47.67	02:03.60	01:48.85	00:58.39
User 3	01:48.63	00:44.32	01:09.76	01:24.43	02:16.13
User 4	01:20.96	00:59.02	02:48.57	02:01.67	00:46.33
User 5	01:16.60	01:06.15	02:47.16	01:59.70	01:37.16
User 6	00:36.08	00:31.87	01:44.31	01:12.15	00:44.26
User 7	00:47.64	00:45.10	02:34.16	01:51.75	00:39.30
User 8	01:15.54	01:03.08	02:12.93	00:59.08	00:52.87
Avg	1:10.70	1:13.79	2:08.10	1:38.02	1:13.18

To have more clear graphical overview over the time performance in these tasks, Figures 4.9, 4.10, 4.11, 4.12, and 4.13 represent graphically the results for the different users in each task alongside a comparison with the respective average time value.

The time results on tasks related to previewing and creating notebooks are particularly satisfying, indicating short times on the main notebook service functions of the implementation. Alongside with the low amount of errors, as observed in the next section, they give us a good metric for the usability of the platform.

4.2.3.3 Errors and Requests for Assistance

When the users were performing the scenario no critical errors were recorded. Under the usability guide that drove this document, critical errors are significant deviations at the completion level from the targets of the scenario. Since this didn't occur through the experimental phase, we can infer one of the usability goals of the document, a level of one hundred percent completion rate in the designated tasks [48]. It should be noted that during the execution of the experiments

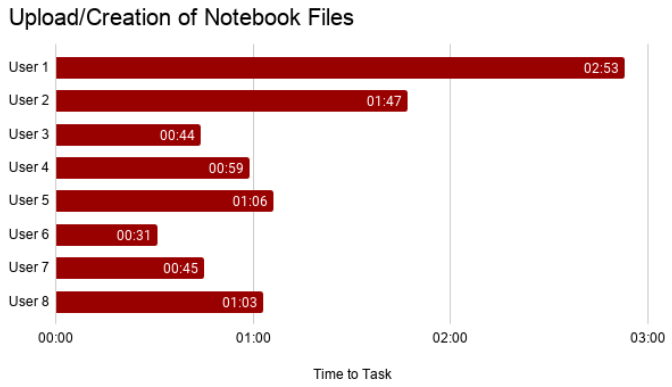

Figure 4.10: Graphical representation of time to task in “Upload/Creation of Notebook Files”.

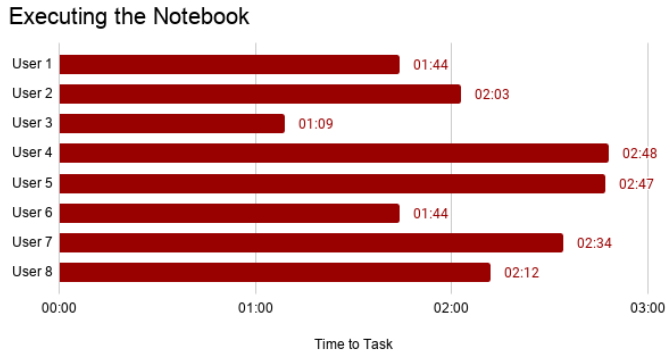

Figure 4.11: Graphical representation of time to task in “Executing the Notebook”.

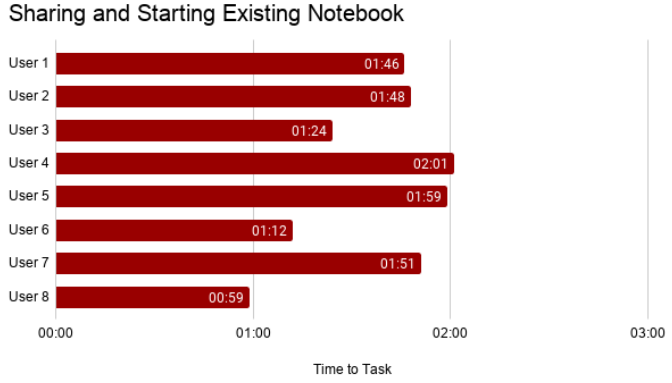

Figure 4.12: Graphical representation of time to task in “Sharing and Starting existing Notebook”.

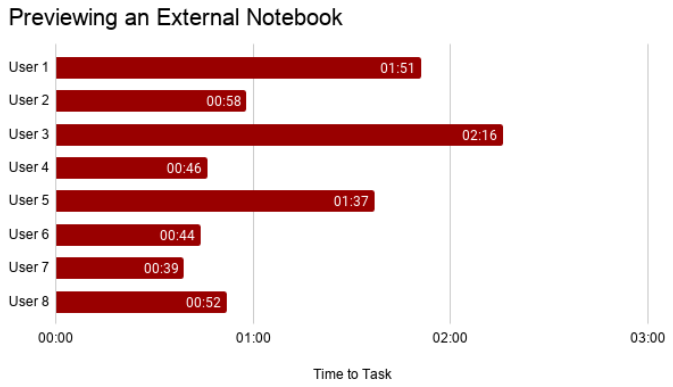

Figure 4.13: Graphical representation of time to task in "Previewing an External Notebook".

one call for assistance was recorded, but it was due to a previously non-verbalized difficulty in understanding the script of the experience and in no way represented an aspect to be considered in the analysis of the implemented work. These tasks however were not free of incidents in which the participants did not complete a scenario in the most optimal form. These are considered non-critical errors, which, as previously referred, were divided into three categories **navigation**, **presentation** and **control usage errors**.

There was only one possible scenario susceptible to **control usage errors**, namely the creation of the notebook, since it required users to input a value for the naming of the notebook object. This however didn't came to fruition so there were no recorded **control usage errors**.

There were however several situations where both **navigation errors** and **presentation errors** occurred. In Figure 4.14 we can see the navigation errors by tasks while in Figure 4.15 we can see the presentation errors.

Tasks one and four, "Creation of a Notebook" and "Sharing and Starting a Notebook", respectively, were where most of the navigation errors occurred, as seen in Figure 4.14. Through this analysis we understood that some aspects within these actions can be improved. In the "Creation of a Notebook" task, most navigation errors were caused by an inability of the users to find the option to "Create a Notebook" within the project context, having returned to the project listing screen or even trying to use the search function to create the notebooks. This led to some suggestions to make the necessary steps to successfully create and use notebooks on the platform more visible. In task number 4 — "Sharing and Starting a Notebook" — the reasoning is similar since the control scheme is the same.

In many ways, the presentations errors seen in Figure 4.15 are similarly caused by the same issues as the navigation errors were, namely by some difficulty in finding the correct objects or misinterpreting some of the information conveyed by the user interface. It should also be noted that on task 5 — "Previewing an External Notebook" — the errors were mostly caused by the change in workflow, from the usage of the drop down menu previously used to start the notebook to a drag and drop mechanism that caused some confusion.

It is interesting to analyse these results and try to extract meaningful insights and conclusions

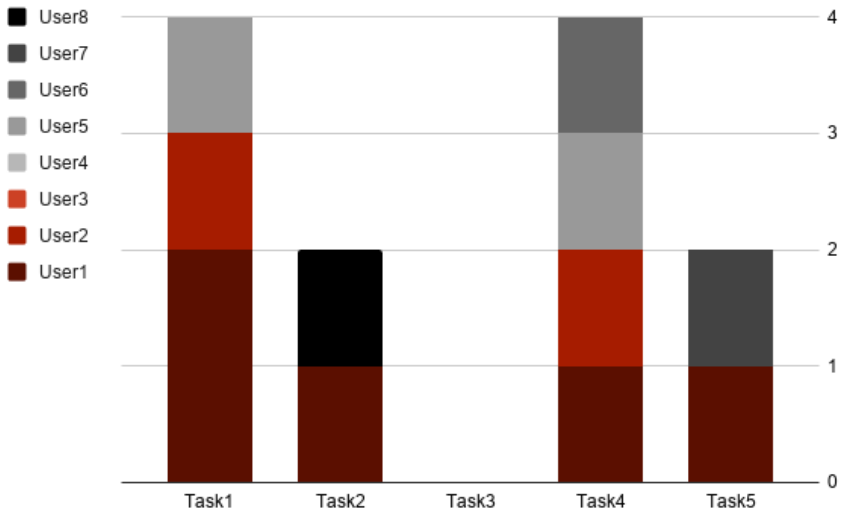

Figure 4.14: Cumulative values for navigation errors.

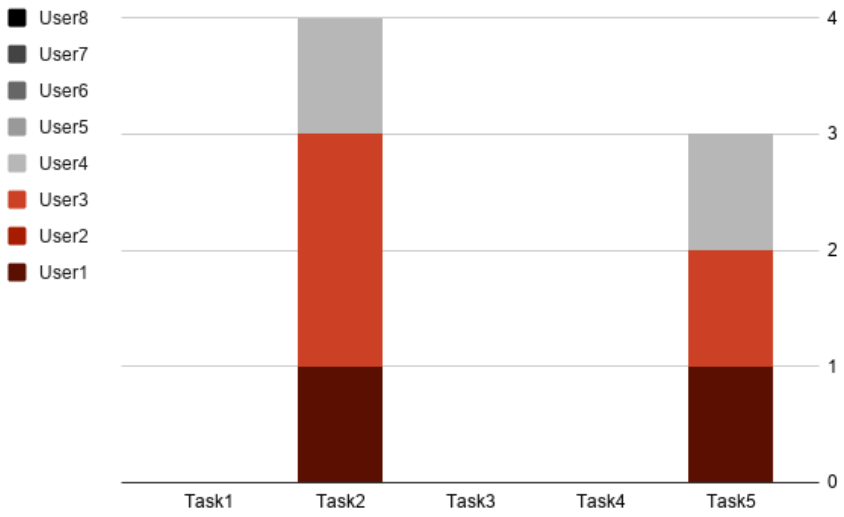

Figure 4.15: Cumulative values for presentation errors.

in order to improve the usability of this work. Ideally these features should be updated and the experimental process repeated in order to improve usability.

4.2.4 System Usability Scale

To conclude this process, participants were prompted to answer a standard usability questionnaire, the System Usability Scale (SUS) [5]. This survey, attached in Appendix B, prompted the subjects to choose an option on a scale from "Strongly Disagree" to "Strongly Agree", to ten statements regarding the implemented features.

1. I think that I would like to use this system frequently
2. I found the system unnecessarily complex
3. I thought the system was easy to use
4. I think that I would need the support of a technical person to be able to use this system
5. I found the various functions in this system were well integrated
6. I thought there was too much inconsistency in this system
7. I would imagine that most people would learn to use this system very quickly
8. I found the system very cumbersome to use
9. I felt very confident using the system
10. I needed to learn a lot of things before I could get going with this system

While these items individually are not meaningful, the SUS is capable of yielding a single number to represent a measure of overall usability of the system [44]. These scores can range from zero to one hundred and will be calculated for each of the SUS questionnaires answered by the participants. In order to calculate this score we must sum contributions from each item, which is variable. For items 1, 3, 5, 7, and 9, the score contribution is the value of the scale position minus one. For items 2, 4, 6, 8 and 10, the contribution is five minus the value of the scale position, as shown in the equation 4.1. In the end of this process, to obtain the overall value on the scale of a hundred we must multiply all the scores by 2.5.

$$\left(\sum_{n=(Item1,3,5,7,9)} n-1\right)+\left(\sum_{n=(Item2,4,6,8,10)} 5-n\right) \tag{4.1}$$

The obtained results from the survey can be seen in table 4.2. The obtained SUS score is hard to interpret if no scale of measurement is provided, but there are however several established ways to measure the scores of the test. In Figure 4.16 different scoring methods are shown [43]. From the presented methods the "percentile" and "acceptability" method are appropriate to extract some conclusions over the obtained results. The percentile method indicates that the average score is

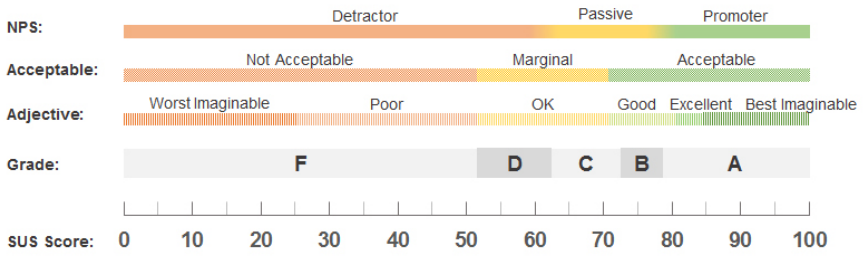

Figure 4.16: Different scoring analysis methods associated with raw SUS scores.

68. This means that obtained that scores can be considered above and below average respectively. The acceptability metric defining what is "acceptable" or "not acceptable, proposes acceptable to correspond to roughly above 70 and unacceptable to below 50 [3]. These two methods support each other. In Figure 4.17 we can see the scores, in red, of the SUS ranked in a graph conveying the percentile and grade method. We can see that 5 out of 8 results score above the average and therefore also on the "acceptable" category according to accessibility ranking. From the other 3 results that score below the average, two are within the 60 percentile and one of them is ranked significantly below, with a score of 37.5. This score indicates that there are still some issues with the user interface and workflow as previously discussed, since this score was directly from one of the users that struggled the most with completing the tasks at hand, but it should also be stated that this user had no previous experience with the platform or with notebooks.

Table 4.2: Results of the System Usability Survey

	User 1	**User 2**	**User 3**	**User 4**	**User 5**	**User 6**	**User 7**	**User 8**
Item 1	4	3	3	3	5	2	3	4
Item 2	3	1	1	1	2	3	1	2
Item 3	4	3	4	4	5	3	3	4
Item 4	1	1	1	1	1	4	2	3
Item 5	4	5	3	5	5	3	3	3
Item 6	3	1	1	1	1	3	3	2
Item 7	5	4	3	5	5	2	2	3
Item 8	3	2	1	1	3	3	2	2
Item 9	4	5	4	4	5	2	3	4
Item 10	1	2	2	2	1	4	1	2
Sum	30	33	31	35	37	15	25	27
SUS Score	75	82.5	77.5	87.5	92.5	37.5	62.5	67.5

4.2.5 User Feedback

The final part of this procedure was a discussion with the subjects about their general opinion on the impact of this work and how it delivered based on its initial goal.

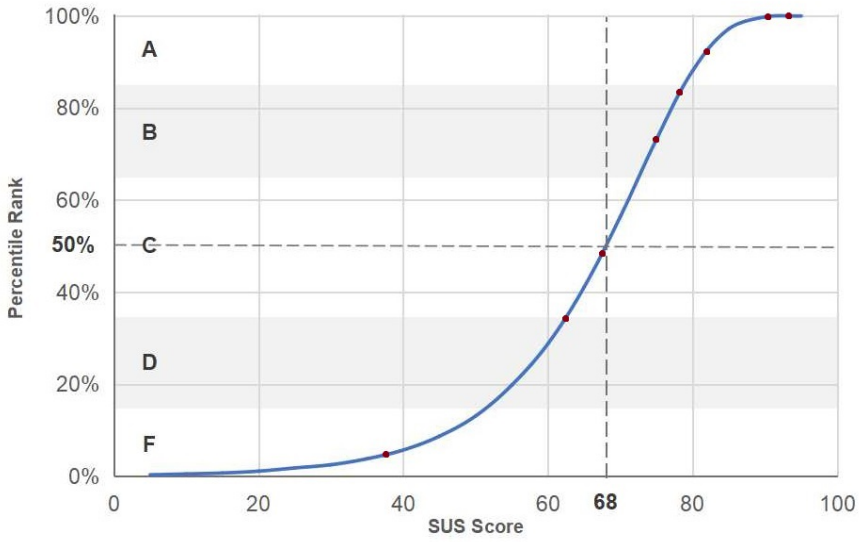

Figure 4.17: SUS scores represented in red against percentile and grade graph

They were asked to fill another short form and finished by having a small personal discussion where suggestions and opinions were registered. This form contained a series of statements reflecting some of the goals of this implementation, and can be consulted in Appendix A.

The statements were the following:

- I believe this implementation brings improvement to the Dendro platform.
- This work improves the processing and data management aspects of data life cycle in the platform.
- This work motivates the use of visualization in the platform.
- This work would brings improvements to the way data can be shared in the platform.
- This work can bring improvements to my workflow when dealing with data.
- I feel more compelled to create visualizations and annotate my work thanks to this implementation.
- This work makes the process of sharing and interpreting datasets more convenient.
- This work can have a significant impact in reproducibility of data.

From the analysis of Figure 4.18 we can infer that generally all the answers were positive and that the implemented work can have a significant impact in the way the platform is used.

Some of the participants took the opportunity provided at the end of the questionnaire to provide some suggestion over the implemented features and some general views over this work.

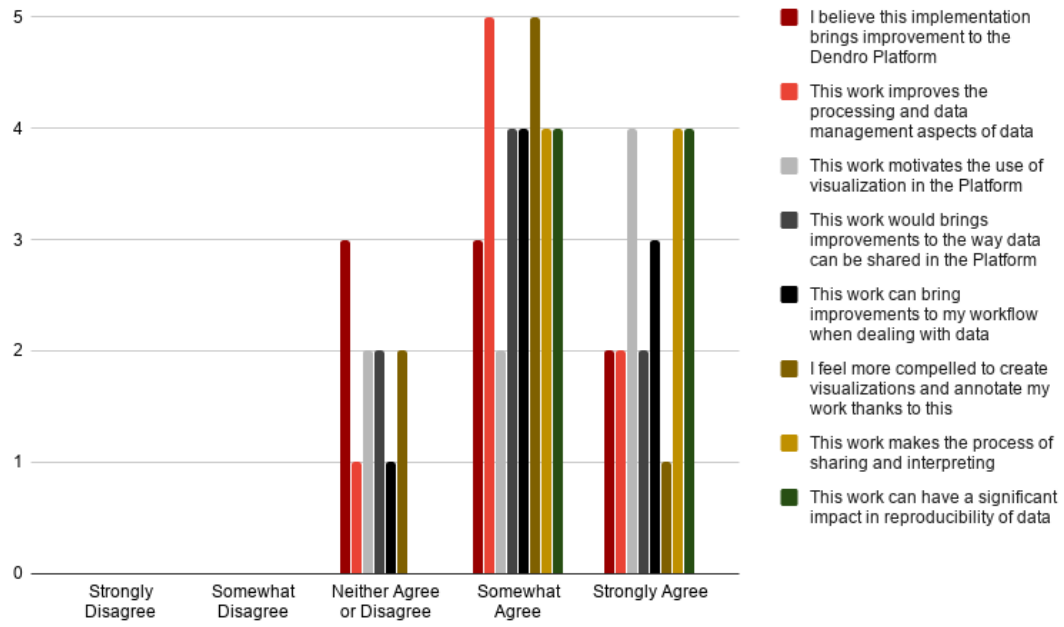

Figure 4.18: Results from the Implementation Appreciation Form

Some of these suggestions linked directly with the preview and visualization features of the notebook service:

"I think the visualization (preview) aspect of the implementation should be more easily accessible, previews for all the existing notebooks in a project would be helpful in order to reduce time spent searching for notebooks."

and

"I would like to see the option to explore notebooks existing in Dendro outside of the projects themselves."

These suggestions relating to the rendering and preview of notebooks are interpreted as a signal of interest and relative success of the achieved implementation, since most subjects found the feature interesting and wanted to see it expanded in other sections of the platform.

Some participants brought forward interesting suggestions relative to the notebook workflow.

"I would like to upload my notebooks and automatically generate notebook directories."

By large the subjects that took part in this questionnaire accompanied the implementation of the notebook service and helped shape this work. Its a good indicator that their level of satisfaction with the features hits some of the goals envisioned for the platform by this work.

There is a general consensus that sharing visualizations and processing methods within the platform has been improved by this work.

To extract a concrete value from Figure 4.18, we can look at table 4.3, where, to obtain a single result for each of the statements, we calculated the average of the stacked results and rounded to the closest factor from "Strongly Disagree" to "Strongly Agree". The obtained results consist in

six statements getting the "Somewhat Agree" classification and two obtaining the highest classification in the scale, "Strongly Agree". These are noticeably good results especially considering that the "This work can have a significant impact in reproducibility of data." statements, the main goal of this implementation, achieved the "Strongly Agree" classification by the participants in the survey.

Table 4.3: Results for satisfaction form in table form

	Strongly Disagree	Somewhat Disagree	Neither Agree or Disagree	Somewhat Agree	Strongly Agree	Average Score	Final Rating
I believe this implementation brings improvement to the Dendro platform	0	0	3	3	2	3,875	Somewhat Agree
This work improves the processing and data management aspects of data life cycle in the platform	0	0	1	5	2	4,125	Somewhat Agree
This work motivates the use of visualization in the platform	0	0	2	2	4	4,25	Somewhat Agree
This work would brings improvements to the way data can be shared in the platform	0	0	2	4	2	4	Somewhat Agree
This work can bring improvements to my workflow when dealing with data	0	0	1	4	3	4,25	Somewhat Agree
I feel more compelled to create visualizations and annotate my work thanks to this implementation	0	0	2	5	1	3,875	Somewhat Agree
This work makes the process of sharing and interpreting datasets more convenient	0	0	0	4	4	4,5	Strongly Agree
This work can have a significant impact in reproducibility of data	0	0	0	4	4	4,5	Strongly Agree

One of the core goals of the implemented work was to increase reproducibility of scientific data, and the researchers that were interviewed in the conduction of this survey agree that some of the essential tools in order to achieve this have been introduced within Dendro.

Chapter 5

Conclusions and Future work

In this final chapter, an overview of all developed work is presented. We reflect on the success of the implementation and the main accomplishments and conclusions achieved in this dissertation. We also provide some suggestions for the further development of this work either being tangible features that weren't implemented or interesting concepts that can possibly be expanded upon.

5.1 Conclusions

When establishing the goals for this dissertation, the fundamental aspect to be demonstrated was that, through the inclusion of web notebooks as a service in data management platforms, we would be able to achieve higher levels of reproducibility as a consequence of pairing data with respective processing code and visualizations. Through the observed results of our experimental method and consultation with the selected expert user we have a positive outlook. Even though the sample is small, most of the users that came across this implementation recognised its potential in improving the workflow and appeal to a more careful treatment of research data. However the concept at discussion is relatively new and therefore only time will tell if it will be successfully accepted and used as a standard. The implementation however was successful in providing tools capable of incorporating processing and visualization with the respective data. In that light, the implementation of this dissertation provides us with positive outlooks on these matters and sets a good basis to expand upon. The state of the art analysis in particular was fundamental towards understanding the importance of solutions encouraging reproducibility. There is a widespread understanding that Open Science benefits will be at the core of a more efficient scientific process [30]. Open Data in particular is where this implementation can have an effect since it seemed to progress at a slower pace than other categories [45]. Web notebooks, with their accessibility and versatility are suggested by many as a solution for this problem, by combining the features of these platforms with the already established scientific data management platforms to bring innovative ways to approach open data [54, 38].

Through the interviews with researchers we can understand that several fields of study use Dendro as a platform. To provide visualization solutions that allow techniques from data monitoring to presentation is essential in order to guarantee that the platform has an appeal for the multiple disciplines of the user groups that work with Dendro. By analysing and implementing solutions that support visualization it became clear how much visualization is capable of impacting scientific research. With the large availability of data and ease of storing very large amounts of data it becomes fundamental to couple structures that hold this data with methods to perform annotation in a more appealing way than plain text.

The experimental process and the user feedback received indicates a positive outlook on the achieved results of this implementation. Around fifty percent of the inquired participants agreed strongly that this implementation facilitates the sharing and interpretation of data. The same number of participants also agree that this work can have a significant impact on the reproducibility of data. The conversations carried out with these participants indicated that the adaptability of the notebook and the interactions with the data itself and visualization are the aspects that approach this implementation's goals of improving reproducibility and encouraging Open Science through the ability to use web notebooks.

This dissertation has built the foundation for a highly versatile platform capable of combining traditional data management with visualization and processing, effectively making data feel lively within Dendro. The full extent to which these capabilities can be explored is still not certain given the modular nature and possibility for expansion of Jupyter Notebook, but the base work has been set and can now be expanded upon.

5.2 Future Work

Different aspects can be improved upon in order to contribute to a more versatile web notebook service. One of the ways this work can be improved upon is by the integration of different visualization libraries with the current established notebook. The current integration of Jupyter Notebook with Dendro allows for a considerable number of visualization solutions to be used but does not guarantees compatibility with some popular libraries that are not native to Jupyter. Gathering a list of some of the popular libraries (based on this state of the art work for instance) and integrating them with the current Jupyter image would strengthen this service.

The synchronization process between the notebook containers and the database can still be improved, specifically when dealing with large projects with a big number of files or with the periodic nature of the synchronization process.

The internal structure of the notebook within Dendro is still limited being very similar in nature to project folders. This hinders the process of cloning or for instance forking notebooks by other projects, making them static within the project they are contained in. It would be interesting to see this feature expanded allowing for different research groups to more easily access each others work.

Dendro is composed of several aspects, for instance the social features, and not all of these aspects are linked to this new service. It would be interesting to allow cross platform notification for successful notebooks or even using these social features as a way to share these notebooks, increasing the reproducibility potential of this implementation.

These are just some aspects that can further improved in order to additionally push towards the vision of improving reproducibility through the usage of web notebooks.

References

[1] Ricardo Carvalho Amorim, João Aguiar Castro, João Rocha da Silva, and Cristina Ribeiro. A comparison of research data management platforms: architecture, flexible metadata and interoperability. *Universal Access in the Information Society*, 16(4):851–862, 2017.

[2] Monya Baker. “Is there a reproducibility crisis? A Nature survey lifts the lid on how researchers view the crisis rocking science and what they think will help”. Nature, vol. 533, no. 7604, p. 452+, 2016.

[3] Aaron Bangor. An Empirical Evaluation of the System Usability Scale: International Journal of Human–Computer Interaction: Vol 24, No 6.

[4] Shannon Bohle. What is E-science and How Should it be Managed? *Spektrum der Wissenshaft*, 2013.

[5] John Brooke. SUS-A quick and dirty usability scale. Technical report.

[6] S. K. Card, J. D. Mackinlay, and B. Schneiderman. Readings in Information Visualization: Using Vision to Think. San Francisco: Morgan Kaufmann. 1999.

[7] Stuart K Card and Jock Mackinlay. "The structure of the information visualization design space," Proceedings of VIZ ’97: Visualization Conference, Information Visualization Symposium and Parallel Rendering Symposium, Phoenix, AZ, USA,pp. 92-99, 1997.

[8] CKAN. Using CKAN: storing data for re-use. https://ckan.org/files/2012/08/OKF-OR12-poster.pdf.

[9] European Comission. Trends for open access to publications. https://ec.europa.eu/info/research-and-innovation/strategy/goals-research-and-innovation-policy/open-science/open-science-monitor/trends-open-access-publications_en.

[10] European Comission. Open Science Monitor Study on Open Science: Monitoring Trends and Drivers. Technical report, 2019.

[11] D3. D3.js - Data-Driven Documents, https://d3js.org.

[12] João Rocha da Silva, Cristina Ribeiro, and João Correia Lopes. Ranking Dublin Core descriptor lists from user interactions: a case study with Dublin Core Terms using the Dendro platform. *International Journal on Digital Libraries*, pages 1–20, 2018.

[13] Paul A David, Ralph Schroeder, William H Dutton, Paul Jeffreys, and Matthijs Den Besten. Will e-Science Be Open Science? Technical report, 2009.

[14] Distill. Distill.Pub, https://distill.pub.

[15] Docker Inc. Docker overview, https://docs.docker.com/engine/docker-overview.

[16] Docker Inc. Empowering App Development for Developers, https://www.docker.com.

[17] Thea Marie Drachen, Ole Ellegaard, Asger Væring Larsen, and Søren Bertil Fabricius Dorch. Sharing Data Increases Citations. *LIBER QUARTERLY*, 26(2):67–82, aug 2016.

[18] Distill Editors. Distill Update 2018. https://distill.pub/2018/editorial-update/.

[19] GoFAIR.org. FAIR Principles - GO FAIR, https://www.go-fair.org/fair-principles.

[20] InfoLab Information Systems Research Group. InfoLab | Information Systems Research Group, https://infolab.fe.up.pt.

[21] Hadley Wickham. Create Elegant Data Visualisations Using the Grammar of Graphics | ggplot2, https://ggplot2.tidyverse.org.

[22] Jeffrey Heer and Ben Shneiderman. Visualization 1 Interactive Dynamics for Visual Analysis A taxonomy of tools that support the fluent and flexible use of visualizations. Technical report.

[23] I. Herman, G. Melancon, and M.S. Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):24–43, 2000.

[24] Observable Inc. Observable. https://beta.observablehq.com/.

[25] Michael Droettboom John Hunter, Darren Dale, Eric Firing. Matplotlib: Python plotting, https://matplotlib.org.

[26] Jupyter Project. Jupyter Notebook. https://jupyter.org/, 2019.

[27] Margaret Rouse. What is a Docker Image and How is it Used?, https://searchitoperations.techtarget.com/definition/Docker-image, 2019.

[28] Bruno Monteiro Marques, João Rocha da Silva, and Tiago Devezas. Visualization in Reproducible Science: A comparative overview of interactive Web Journals and computational notebooks. pages 7–10. 2019 14th Iberian Conference on Information Systems and Technologies (CISTI), 2019.

[29] Wolfram Mathematica. Wolfram Mathematica: Computação técnica moderna, https://www.wolfram.com/mathematica.

[30] Marcia McNutt. Improving Scientific Communication. *Science*, 342(6154):13–13, oct 2013.

[31] Ingeborg Meijer and Tung Tung Chan. The Netherlands' Plan on Open Science. Technical report, European Commission, 2018.

[32] Sebastian Neumaier, Jürgen Umbrich, and Axel Polleres. Challenges of mapping current CKAN metadata to DCAT. Technical report.

[33] Jakob Nielsen. Why You Only Need to Test with 5 Users. *Nielsen Norman Group*, mar 2000.

[34] Chris Olah and Shan Carter. Attention and Augmented Recurrent Neural Networks. *Distill*, Volume 1, sep 2016.

[35] Astrid Orth, Nancy Pontika, and David Ball. FOSTER's Open Science Training Tools and Best Practices. In: Positioning and Power in Academic Publishing: Players, Agents and Agendas: Proceedings of the 20th International Conference on Electronic Publishing (Loizides, Fernando and Schmidt, Birgit eds.). pages 135–141, 2016.

[36] Claus Pahl, Antonio Brogi, Jacopo Soldani, and Pooyan Jamshidi. Cloud Container Technologies: a State-of-the-Art Review. *IEEE Transactions on Cloud Computing*, 7161(c):1–14, 2017.

[37] Jeffrey M. Perkel. Data visualization tools drive interactivity and reproducibility in online publishing. *Nature 2018 554:7690*, jan 2018.

[38] Bernadette M. Randles, Irene V. Pasquetto, Milena S. Golshan, and Christine L. Borgman. Using the Jupyter Notebook as a Tool for Open Science: An Empirical Study. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries*, pages 1–2. IEEE, jun 2017.

[39] João Rocha Da Silva, Cristina Ribeiro, and João Correia Lopes. Ontology-based multi-domain metadata for research data management using triple stores, IDEAS '14: Proceedings of the 18th International Database Engineering Applications Symposium, Pages 105–114, july 2014.

[40] Margaret Rouse. What is the Data Lifecycle, https://whatis.techtarget.com/definition/data-life-cycle, 2017.

[41] Adam Rule, Amanda Birmingham, Cristal Zuniga, Ilkay Altintas, Shih-Cheng Huang, Rob Knight, Niema Moshiri, Mai H. Nguyen, Sara Brin Rosenthal, Fernando Pérez, and Peter W. Rose. Ten Simple Rules for Reproducible Research in Jupyter Notebooks. Technical report, 2018.

[42] Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. Vega-Lite: A Grammar of Interactive Graphics. 2017.

[43] Jeff Sauro. MeasuringU: 5 Ways to Interpret a SUS Score. https://measuringu.com/interpret-sus-score.

[44] U.S. Department of Health & Human Services. About Us | Usability.gov. https://www.usability.gov/about-us.

[45] Elta Smith and Salil Gunashekar. Open Science Monitoring The team at RAND Europe included: Impact Case Study Zenodo. Technical report.

[46] Traefik. Traefik, The Cloud Native Edge Router | Containous, https://containo.us/traefik.

[47] University of Cambridge. How much do publishers charge for Open Access? | Open Access, https://www.openaccess.cam.ac.uk/paying-open-access/how-much-do-publishers-charge-open-access.

[48] U.S. Department of Health & Human Services. Methods https://www.usability.gov/how-to-and-tools/methods/index.html.

[49] Usability.Gov. Improving the User Experience - Use Cases. https://www.usability.gov/how-to-and-tools/methods/use-cases.html.

[50] J.J. van Wijk. The Value of Visualization. In *IEEE Visualization 2005 - (VIS'05)*, pages 11–11. IEEE.

[51] Vega Project. Examples gallery Vega. https://vega.github.io/vega/examples/.

[52] Junpeng Wang, Subhashis Hazarika, Cheng Li, and Han Wei Shen. Visualization and Visual Analysis of Ensemble Data: A Survey. *IEEE Transactions on Visualization and Computer Graphics*, PP(8):1, 2018.

[53] Mark D Wilkinson. The FAIR Guiding Principles for scientific data management and stewardship. 2016.

[54] Jack Zentner and Tom McDermott. Web notebooks as a knowledge management tool for system engineering trade studies. In *11th Annual IEEE International Systems Conference, SysCon 2017 - Proceedings*, 2017.

Appendix A

Implementation Overview Questionnaire

Subject Profiles

This form is to be filled by the facilitator/data logger

1. Subject ID

_____

2. How would you describe your level of experience with Web Notebooks?

Marcar apenas uma oval.

	1	2	3	4	5	
Not familiar with the concept	◯	◯	◯	◯	◯	Daily Usage

3. How would you describe your level of experience with Data Management Platforms?

Marcar apenas uma oval.

	1	2	3	4	5	
Not familiar with the concept	◯	◯	◯	◯	◯	Daily Usage

4. How familiar are you with data processing tools?

Marcar apenas uma oval.

	1	2	3	4	5	
Not familiar with the concept	◯	◯	◯	◯	◯	Daily Usage

General Appreciation

5. I believe this implementation brings improvement to the Dendro Platform.

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

6. This work improves the processing and data management aspects of data life cycle in the Platform.

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

7. This work motivates the use of visualization in the Platform.

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

8. This work would brings improvements to the way data can be shared in the Platform.

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

9. This work can bring improvements to my workflow when dealing with data.

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

10. I feel more compelled to create visualizations and annotatte my work thanks to this implementation.

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

11. This work makes the process of sharing and interpreting datasets more convenient.

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

12. This work can have a significant impact in reproducibility of data.

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

13. Improvements/Suggestions for implementations.

Appendix B

System Usability Scale Form

System Usability Scale

This is a form using defined metrics in order to obtain a System Usability Scale

1. I think that I would like to use this system frequently

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

2. I found the system unnecessarily complex

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

3. I thought the system was easy to use

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

4. I think that I would need the support of a technical person to be able to use this system

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

5. I found the various functions in this system were well integrated

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

6. I thought there was too much inconsistency in this system

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

7. I would imagine that most people would learn to use this system very quickly

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

8. I found the system very cumbersome to use

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

9. I felt very confident using the system

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

10. I needed to learn a lot of things before I could get going with this system

Marcar apenas uma oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

Appendix C

Introductory Guide to System Testing Procedure

Welcome and Purpose

Thank you so much for coming in today. Before we begin I wanted to give you some information about what you will be performing in this test session and proceed to give you some time to ask any questions you find appropriate.

In this session we are asking you to serve as an evaluator of a set of website functionalities by completing a number of tasks. The goal is to determine how easy or difficult you find the usage of the website and these features.
Afterwards recording a general opinion over the value of the system.
I am here to record your reactions and comments about the website during the tasks you will be performing.

Test Facilitator's Role

During this session, you are free to think aloud as you work to complete the tasks. I will not be able to offer any suggestions or hints, but from time to time, I may ask you to clarify what you have said or ask you for information on what you were looking for or what you expect to have happen.

Test Participant's Role

Today I am going to be asking you to perform some actions on the website and tell me how easy or difficult it was to perform this tasks. These activities have as purpose to evaluate the ease of use of the implemented features.
There are no right or wrong answers. If you have any questions, comments or areas of confusion while you are working, please let me know.
If you ever feel that you are lost or cannot complete a task with the information that you have been given, please let me know. I will ask you what you might do in a real-world setting and then either put you on the right track or move you on to the next scenario.
As you use the site, please do so as you would at home or your office. I would ask that you to try work through the tasks based on what you see on screen, but if you reach a point where you are not sure where or how to find something please inform me.
Your name will not be associated or reported with data or findings from this evaluation.
I may ask you other questions as we go and we will have wrap up questions at the end.

Do you have any questions before we begin?