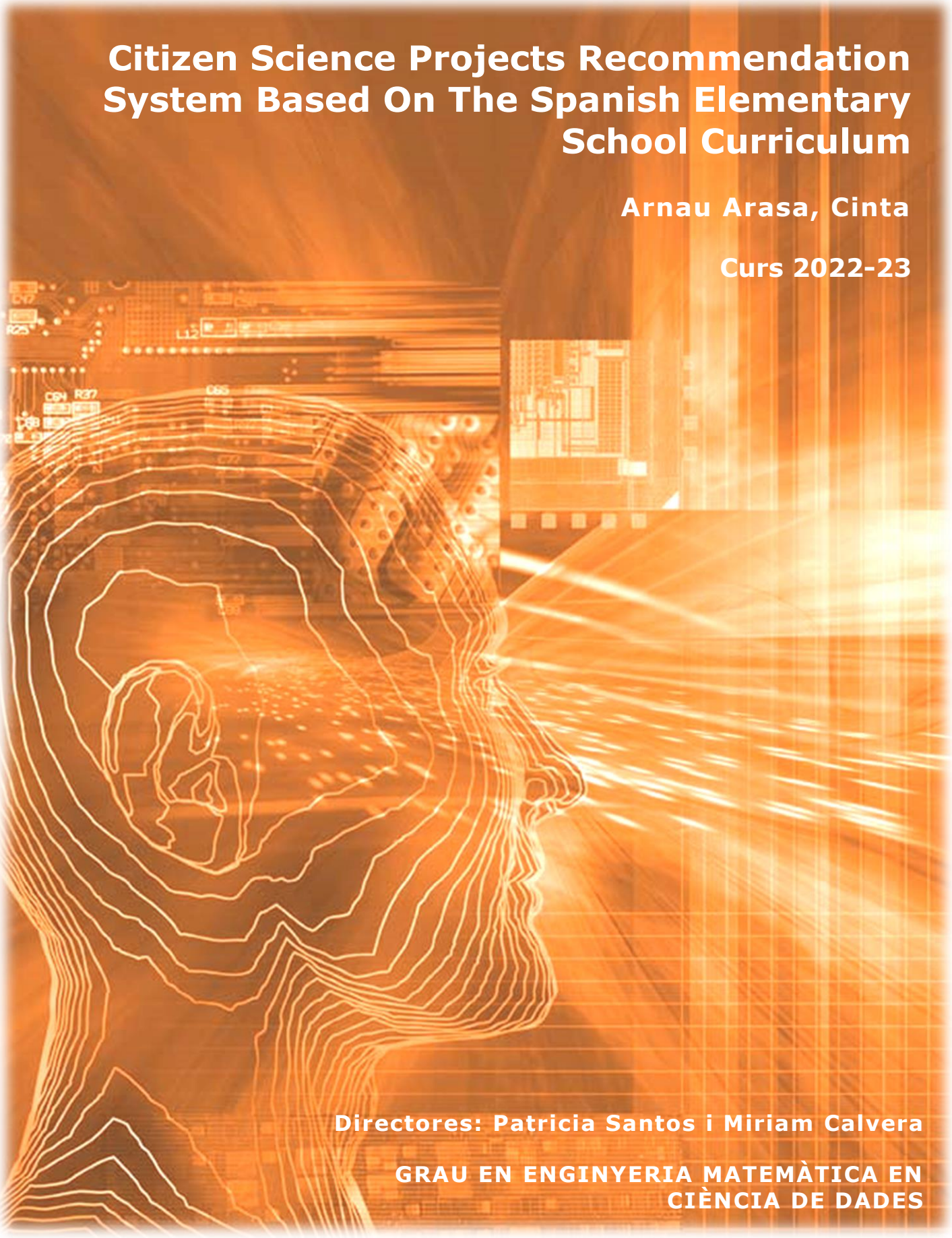
Citizen Science Projects Recommendation System Based On The Spanish Elementary School Curriculum

Arnau Arasa, Cinta

Curs 2022-23



Directores: Patricia Santos i Miriam Calvera

GRAU EN ENGINYERIA MATEMÀTICA EN CIÈNCIA DE DADES

upf. Universitat Pompeu Fabra Barcelona
Escola d'Enginyeria

Treball de Fi de Grau

[Pàgina en blanc]

[S'han marcat els apartats del treball que haurien d'anar en pàgina senar o dreta de la publicació (situació de pàgines que ajuda a estructurar i a donar prioritat formal als diferents apartats).

Hi ha apartats on no s'indica res. En aquests cas l'estudiant pot triar la situació, en pàgina parell o senar, segons li convingui per al compaginat final.

Es recomana que les pàgines blanques no portin número de foli (es compten, però el número no s'imprimeix)]

Si s'imprimeix a simple cara no s'han de deixar pàgines en blanc

To my younger sister Núria
who I love the most

[Pàgina en blanc]

Acknowledgements

Text dels agraïments [12 punts]

[Pàgina en blanc]

Abstract

Resum del treball en la llengua que es presenti [12 punts]
Extensió màxima recomanada: 150 paraules

Resumen [en una 2a llengua. Ex. Resumen]

Resum del treball en una llengua diferent a la utilitzada [12 punts]
Extensió màxima recomanada: 150 paraules

Resum [en una 3a llengua. Ex. Abstract]

Resum del treball en una llengua diferent a la utilitzada [12 punts]
Extensió màxima recomanada: 150 paraules

*[Aquest apartat, **sempre** hauria de començar en pàgina senar, pàgina dreta de la publicació]*

[Pàgina en blanc]

TABLE OF CONTENTS

[Taula de contingut del treball, parts en què està dividida] [12 punts]

[Aquest apartat també ha de començar en pàgina senar]

- Text marked with this color indicates that it is a part of the project that is still not done and that part will be documented later. It is marked to make sure it is not left undone!
- References are marked with this color to indicate that the number will be later changes. There are previous sections still not done and therefore the number of the reference will change order. It is marked to make sure it is not left unchanged!

Table of figures [opcional]

Llista de taules [opcional]

1. INTRODUCTION

1.1 Títol de l'apartat [arial 14 punts]

Text de l'apartat [12 punts]

a) Títol de la subdivisió [14 punts]

Text de la subdivisió a) [12 punts]

b) Títol de la subdivisió [14 punts]

Text de la subdivisió b). Nota a peu de pàgina [1] [12 punts]

[Cada capítol ha de començar en pàgina senar]

[1] Text de la nota al peu de pàgina [10 punts]

2. STATE OF THE ART

2.1 Títol de l'apartat [14 punts]

Text de l'apartat [12 punts]

a) Títol de la subdivisió [14 punts]

Text de la subdivisió a) [12 punts]

b) Títol de la subdivisió [14 punts]

Text de la subdivisió b). Nota a peu de pàgina [2] [12 punts]

[Cada capítol ha de començar en pàgina senar]

[2] Text de la nota al peu de pàgina [10 punts]

3. DESIGN AND IMPLEMENTATION

In this section are shown the different tools and platforms containing citizen science projects used in the making of this project. The platforms used are the Spanish platform "Observatorio de la Ciencia Ciudadana en España"[3] and the Barcelonian platform "Oficina de la Ciència Ciutadana"[4]. It is explained the data structure of each of these platforms and the extraction process of all the data related to the citizen science projects.

3.1 Environment set-up and selection of tools

3.1.1) Tools Used

In order to extract the data from the two platforms of Citizen Science projects and create the recommendation system various tools have been used throughout the entirety of the thesis process. The following descriptions of these tools aim to clarify why they are necessary and demonstrate their application within the project.

3.1.1.1) Python

Python[5] is a high-level programming language that is widely used for general-purpose programming. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python's versatility and ease of use have made it a go-to language for a wide range of applications, including web development, data analysis, machine learning, artificial intelligence, scientific computing, and automation [1]. For this reason, and given the fact that Python has been the most used programming language throughout the degree, it has been the chosen language for the Citizen Science projects' extraction and the building of the recommendation system.

The libraries that have been used in the scripts (both the project extraction and the recommendation system) are the following:

- Pandas[6]: "It is a powerful open-source Python library widely used for data manipulation, analysis, and exploration". It provides highly efficient data structures and data analysis tools, making it an essential tool for working with structured data, creating DataFrame objects to manipulate data with integrated indexing, and much more functionalities to work with data sets [2].
- Request[7]: "It is a popular tool used for making HTTP requests and interacting with web services. It provides a convenient and user-friendly interface to send HTTP requests, handle responses, and perform various operations related to web communication" [3].
- BeautifulSoup[8]: "Python library used for web scraping and parsing HTML or XML documents" [4].
- (…)

[3] "Observatorio de la Ciencia Ciudadana en España" (2023, May 20) https://ciencia-ciudadana.es/.
[4] "Oficina de la Ciència Ciutadana" (2023, May 20) https://www.barcelona.cat/barcelonaciencia/es/.
[5] Python Website (2023, May 20) https://www.python.org/.
[6] Pandas Documentation (2023, May 20) https://pandas.pydata.org/docs/.
[7] Request Documentation (2023, May 20) https://docs.python-requests.org/en/latest/.
[8] BeautifulSoup Documentation (2023, May 20) https://www.crummy.com/software/BeautifulSoup.

3.1.1.2) Jupyter Notebook

"Jupyter Notebook[9] is an open-source web-based interactive computing environment widely used for data analysis, visualization, and prototyping" [5]. It supports many programming languages (including Python) and allows for interactive data analysis and visualization, enhancing the exploratory data analysis process. Since Jupyter Notebook has been the most used framework throughout the degree, it has been the chosen one for the creation

3.1.1.3) BeautifulSoup

"BeautifulSoup is a popular Python library used for web scraping and parsing HTML or XML documents" [4]. It provides a convenient way to extract and navigate data from web pages by simplifying the process of locating and manipulating elements within the document structure. BeautifulSoup can be combined with other Python libraries, such as requests, to fetch web page content and then parse and extract the desired information.

3.1.1.4) GitHub

"GitHub[10] is a web-based platform for version control and collaborative software development". It provides a centralized hub where developers can store, manage, and collaborate on their code repositories [6]. A GitHub repository[11] has been created to store and manage all the scripts with the corresponding code of this thesis.

3.1.2) Jupyter Notebook Scripts

In order to extract all the information about the Spanish and Catalan Citizen Science projects, the script "projects_extraction.ipynb" has been generated. All the information is extracted using the Python library BeautifulSoup as it will be later explained.

The other script named "recommendation_system.ipynb" (…).

These Jupyter notebook scripts can be found in the Github repository so that anyone interested in Citizen Science or any teacher wanting to use the recommendation system can consult them (see Appendix X to consult the locations of the scripts).

3.2 Spanish platform: "Observatorio de la Ciencia Ciudadana en España"

3.2.1) Platform data structure

(The Spanish platform "Observatorio de la Ciencia Ciudadana" is right now under maintenance. Once it is fixed, this part will be redacted).

3.2.1) Extracting the data from the Spanish platform

In order to extract all the information from the Spanish platform, it is needed to create a crawler. The chosen pyhton library to do the web scraping is BeautifulSoup, which will make it possible to get the desired data throughout the examination of the website

[9] Jupyter Notebook Website (2023, May 20) https://jupyter.org/.
[10] GitHub Website (2023, May 20) https://docs.github.com/en.
[11] GitHub repository https://github.com/Cintaa1223/TFG.

elements. The process to do so in a Jupyter Notebook environment, as it is done in the "projects_extraction.ipynb" script, is the following:

1. Install and import the BeautifulSoup library along with the other needed libraries pandas and request.

```
!pip install BeautifulSoup
!pip install requests
!pip install pandas
```

Figure 1. Installing Python libraries.

```
import requests
from bs4 import BeautifulSoup
import pandas as pd
```

Figure 2. Code to import libraries.

2. Sending an HTTP request to the web page by using the 'requests.get()' function to send a GET request to the URL of the Spanish website. The response from the server can be used to extract the HTML content of the page.

```
url = "https://ciencia-ciudadana.es/proyecto-cc/"
web_name = 'Observatorio de la Ciencia Ciudadana en España'

# Adding headers to the rquest
headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) '
                         'AppleWebKit/537.36 (KHTML, like Gecko) '
                         'Chrome/58.0.3029.110 Safari/537.36'}

response = requests.get(url, headers=headers)
```

Figure 3. Code to send HTTP request.

3. Parsing the HTML content. The HTML content obtained from the web page is passed to the BeautifulSoup constructor to create a BeautifulSoup object, which allows to navigate and search through the HTML structure of the page.

```
soup = BeautifulSoup(response.content, "html.parser")
```

Figure 4. Code to parse HTML content.

4. Extracting data. This involves finding specific HTML elements such as tags, classes, or IDs, and accessing their attributes or text content. In order to do so, methods like 'find()' or 'find_all()' are used to locate and extract the desired data.

   To see the HTML structure of the webpage: "Ajustes" → "Más herramientas" → "Herramientas para desarrolladores"[12].

   - First of all, the home page of the Spanish platform contains all the projects which have to be accessed in order to extract the needed information. To do so, we first find where the URL to each project is found in the HTML structure and create a list that contains all the links to the citizen science projects.

```
# Find all elements with class name "underline"
underline_elements = soup.find_all('img', {'decoding': 'async'})

links = []
# Extract the links from the parent elements
for element in underline_elements:
  parent_a_tag = element.find_parent('a')
  if parent_a_tag and 'href' in parent_a_tag.attrs:
    link = parent_a_tag['href']
    links.append(link)
```

Figure 5. Code to extract projects' URLs.

   - Now we access each of the projects' URLs stored in the links array the same way as described in steps 2 and 3. The array can be iterated to get all the necessary information of each project. This needed data includes the following

[12] It can also be done by either right-clicking on any part of the webpage and selecting "Inspect" or by clicking 'fn'+'f12'.

fields: *'Project Name', 'Project Link', 'Project Scope', 'Project Goal', 'Project Description', 'Project Entity/Scientist', 'How To Join', 'Necessary Equipment', 'Initial Date', 'Final Date', 'Public Type', 'Location (Province)', 'Number of Participants', 'Results', 'Link to Results', 'Project Impact', 'Why Using CC?', 'Citizen Science Web Name', 'Citizen Science Web Link'*.
Given that each information to be extracted follows the same HTML structure, the function 'get_complete_section(project_soup, dtbf)' has been created to make it simple.

```
def get_complete_section(proj_soup, dtbf):
  proj_seg = proj_soup.find_all('div', {'class': 'tb-field', 'data-toolset-blocks-field': dtbf})
  return ''.join([seg.text for seg in proj_seg])
```

Figure 6. Function get_complete_section(proj_soup, dtbf).

```
def get_project_info1(project_link, main_url, main_name):
  proj_info = ['', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '']

  response = requests.get(project_link, headers=headers)
  proj_soup = BeautifulSoup(response.content, "html.parser")
  proj_title = proj_soup.find('h1', {'class': 'entry-title'}).text
  proj_scope = get_complete_section(proj_soup, "28515d8ce1fa37a6527af15754983e83")
  proj_goal = get_complete_section(proj_soup, "ba74407e8b20cac888e283e8576140f9")
  proj_desc = get_complete_section(proj_soup, "7147dde37d9b86b4d4a2dc89b9c12945")
  proj_entity = get_complete_section(proj_soup, "486908e9509d0a139e234cbfb7f8f47d")
  proj_join = get_complete_section(proj_soup, "687686b4b9d6f02068b541e6bc6f2812")
  proj_equip = get_complete_section(proj_soup, "cbcd66c16990127ea3c06cebc87c17f0")
  proj_ini = get_complete_section(proj_soup, "3d66c4d00ab960c97a58ac752fe406f4")
  proj_end = get_complete_section(proj_soup, "9a00b649618c1cd0c0cedd6cfc386f09")
  proj_public = get_complete_section(proj_soup, "e9e0eef27b43baa290bdc058d0ec6cee")
  proj_loc = get_complete_section(proj_soup, "05399f19f4997ca4dc2a1f1770db2d80")
  proj_amt_part = get_complete_section(proj_soup, "5ee97fadc3c34313fff123d0cba66a1a")
  proj_results = get_complete_section(proj_soup, "b064a4637177bd64d367dc70864be32c")

  proj_link_res = proj_soup.find_all('div', {'class': 'tb-field', 'data-toolset-blocks-field': "c5d8b6598ffd2835065c71792c189772"})
  proj_link_res = [seg.text for seg in proj_link_res][0]

  proj_impact = get_complete_section(proj_soup, "4297f88ef1601b65151fcf8fc6fccadd")
  proj_useCC = get_complete_section(proj_soup, "974e65ee4de5d532951d366fa43dc400")
```

Figure 7. Code to extract all the necessary fields of a project.

```
for link in links:
  get_project_info1(link, url, web_name)
```

Figure 8. Code to iterate through the links array.

5. Storing the extracted data in a pandas DataFrame. In the same function 'get_project_info1()' as in *Figure 7*, all the fields extracted are stored in a dictionary so that it can be added as a new row to the created DataFrame 'df1'.

```
# Create an empty DataFrame with specified columns
df1 = pd.DataFrame(columns=['Project Name', 'Project Link', 'Project Scope', 'Project Goal', 'Project Description',
                            'Project Entity/Scientist', 'How To Join', 'Necessary Equipment', 'Initial Date',
                            'Final Date', 'Public Type', 'Location (Province)', 'Number of Participants', 'Results',
                            'Link to Results', 'Project Impact', 'Why Using CC?', 'Citizen Science Web Name',
                            'Citizen Science Web Link'])
```

Figure 9. Creation of the DataFrame to store the Spanish platform projects.

```
# Create a dictionary with the values for the new row
new_row = {
'Project Name': proj_title,
'Project Link': project_link,
'Project Scope': proj_scope,
'Project Goal': proj_goal,
'Project Description': proj_desc,
'Project Entity/Scientist': proj_entity,
'How To Join': proj_join,
'Necessary Equipment': proj_equip,
'Initial Date': proj_ini,
'Final Date': proj_end,
'Public Type': proj_public,
'Location (Province)': proj_loc,
'Number of Particpiants': proj_amt_part,
'Results': proj_results,
'Link to Results': proj_link_res,
'Project Impact': proj_impact,
'Why Using CC?': proj_useCC,
'Citizen Science Web Name': main_name,
'Citizen Science Web Link': main_url}
```

Figure 8. Storing project fields in a dictionary.

```
# Add the new row to the DataFrame
df1.loc[len(df1)] = new_row
```

Figure 9. Adding dictionary to DataFrame.

3.3 Barcelonian platform: “Oficina de la Ciència Ciutadana”

3.2.1) Platform data structure

(…)

3.2.1) Extracting the data from the Barcelonian platform

(…)

[Cada capítol ha de començar en pàgina senar]

4. RECOMMENDATION SYSTEM

4.1 Analysis of the Spanish elementary school curriculum

4.1.1) Key Competences

(…)

4.1 Creation of the Recommendation System

5. RESULTS AND CONCLUSIONS

6. FUTURE WORK

7. BIBLIOGRAPHY

1. Python Software Foundation. (n.d.). Python. Retrieved May 20, 2023, from https://www.python.org/.
2. Pandas Documentation. (n.d.) Retrieved May 20, 2021, from https://pandas.pydata.org/docs/.
3. Requests Documentation. (n.d.). Retrieved May 20, 2023, from https://docs.python-requests.org/en/latest/.
4. BeautifulSoup Documentation. (n.d.). Retrieved May 20, 2021, from https://www.crummy.com/software/BeautifulSoup/bs4/doc/.
5. Project Jupyter. (n.d.). Jupyter Notebook. Retrieved May 20, 2023, from https://jupyter.org/.
6. GitHub. (n.d.). About GitHub. Retrieved May 20, 2023. from https://github.com/about.

Cal documentar les fonts bibliogràfiques utilitzades amb un format de citació estàndard:

http://guiesbibtic.upf.edu/tic/tfg (apartat ***Presentació del treball*****)**

La Biblioteca de la UPF ofereix el gestor de bibliografies Mendeley, que us permet crear la vostra base de dades personal de referències bibliogràfiques en línia, importar referències automàticament des de diferents recursos d'informació, extreure llistes i generar bibliografies en diferents estils de citació, i incorporar les citacions i llistes de bibliografia als vostres documents de text

Gestor de bibliografies Mendeley

Exemple de cita bibliogràfica [12 punts]

Heery, M. "Organització de la biblioteca: Repàs d'estructures." *Item: Revista de biblioteconomia i documentació 23.2 (1998): 8-15.*

Text de la cita bibliogràfica [12 punts]

[Aquest apartat ha de començar en pàgina senar]

7. APPENDICES