Virtual Editions in the LdoD Archive using Crowdsourcing and Gamification

Gonçalo Medeiros Serapicos Montalvão Marques

Thesis to obtain the Master of Science Degree in Information Systems and Computer Engineering

Supervisors: Ph.D. António Manuel Ferreira Rito da Silva
Ph.D. Manuel José de Freitas Portela

Examination Committee

Chairperson: Ph.D. João Emílio Segurado Pavão Martins
Supervisor: Ph.D. António Manuel Ferreira Rito da Silva
Member of the Committee: Ph.D. Rui Filipe Fernandes Prada

November 2018
Acknowledgments

I would like to thank my mother, father, brother, grandparents, aunt and stepmother for all of their support, encouragement, guidance and patience. I would also like to acknowledge all my friends and colleagues that tested my prototype, gave me suggestions and helped me through this work and in my life. To each and every one of you – Thank you.

I would like to express my deep gratitude to Prof. Antônio Rito da Silva for his valuable and constructive suggestions during the planning and development of this research work and for believing in me and this project. I also wish to express my appreciation to Prof. Manuel Portela for all the help.
Abstract

Today’s Digital Age has changed our way of life, influenced our relationships, our culture and much more. An important breakthrough to this age is the World Wide Web, which in turn caused the emergence of new fields, one of them being the Digital Humanities (DH). Crowdsourcing has emerged with the Web and has become an important sourcing alternative in business as well as in Digital Humanities. Besides crowdsourcing, another process that has risen is Gamification, which focuses in increasing user engagement. This thesis seeks to harness the crowdsourcing and gamification potential, from the (DH) perspective, specifically in the context of the LdoD Archive. The LdoD Archive is a collaborative digital archive of the Book of Disquiet by Fernando Pessoa. It allows users to read and explore the book in a digital way, such as through the creation of Virtual Editions. Therefore the goal of this thesis is to enhance the user experience in the LdoD Archive using crowdsourcing and gamification techniques.

Keywords

Crowdsourcing; Gamification; Fernando Pessoa; LdoD Archive; Book of Disquiet; Digital Humanities; Social Software;
Resumo

A Era Digital mudou o nosso modo de vida, influenciou as nossas relações, a nossa cultura e muito mais. Um importante avanço nesta era é a World Wide Web, que por sua vez causou o surgimento de novos campos, sendo um deles as Humanidades Digitais. Crowdsourcing surgiu com a Web e tornou-se uma alternativa importante no negócio de sourcing assim como nas Humanidades Digitais. Além de crowdsourcing, outro processo que cresceu foi a Gamificação, que se foca em aumentar a interacção com o utilizador. Esta tese procura utilizar o potencial de crowdsourcing e gamificação, através da perspectiva das Humanidades Digitais, especificamente no contexto do Arquivo do LdoD. O Arquivo do LdoD é um arquivo digital colaborativo do Livro do Desassossego do autor Fernando Pessoa. Este arquivo permite aos utilizadores ler e explorar o livro numa forma digital, tal como a criação de Edições Virtuais. Portanto o objectivo desta tese é melhorar a experiência dos utilizadores do Arquivo do LdoD usando técnicas de crowdsourcing e gamificação.

Palavras Chave

Crowdsourcing; Gamificação; Fernando Pessoa; Arquivo do LdoD; Livro do Desassossego; Humanidades Digitais; Software Social;
## Contents

1 Introduction ............................................. 1
   1.1 Motivation ........................................... 2
   1.2 Goals ................................................. 3
   1.3 Document Outline ..................................... 4

2 Related Work ........................................... 5
   2.1 The LdoD Archive ..................................... 6
   2.2 Architecture of the LdoD Archive ......................... 6
   2.3 Virtual Editions in the LdoD Archive ...................... 9
   2.4 Digital Humanities ................................... 9
   2.5 Crowdsourcing ....................................... 10
      2.5.1 Dissecting crowdsourcing through four pillars ....... 11
   2.6 Gamification ......................................... 12
   2.7 Crowdsourcing in the Digital Humanities .................. 14
   2.8 Gamification in Crowdsourcing ............................ 17
      2.8.1 Gamification Recommendations ....................... 18
   2.9 Gamification in Digital Humanities ....................... 19
   2.10 A case study: Transcribe Bentham ......................... 20
   2.11 ESP game: An analysis ................................ 21
      2.11.1 Motivation ........................................ 21
      2.11.2 Description ....................................... 22
      2.11.3 Implementation .................................... 23
      2.11.4 ESP Game Evolution and Present ..................... 24

3 Problem Statement ...................................... 26
   3.1 Virtual Editions ....................................... 27
   3.2 Crowdsourcing and the LdoD Archive ....................... 27
   3.3 Gamification and the LdoD Archive ........................ 29
   3.4 The LdoD Archive and the ESP game ....................... 29
## List of Figures

2.1 Model of the LdoD Archive [1]. ............................................................ 7
2.2 Model of the Virtual Dimension [1]. .................................................... 7
2.3 LdoD Archive Architecture [1]. .............................................................. 8
2.4 New Zealand’s National Library Digital Content Life Cycle model [2]. .... 17
2.5 Geiger & Schader Archetypes of Crowdsourcing Systems [3]. ............... 18
2.6 Example of players agreeing on a guess [4]. ......................................... 22
2.7 ESP game interface [4]. ................................................................. 24
2.8 Current Google Image Labeler. ............................................................ 25

4.1 Game interface of round 1. ................................................................. 34
4.2 Game interface of round 2. ................................................................. 34
4.3 Game interface of round 3. ................................................................. 35

5.1 Enriched UML Class Diagram of the LdoD Archive with Classification Game entities 40
5.2 State machine that illustrates the synchronization flow of a game. .......... 44
5.3 New game interface features in the LdoD Archive. ............................. 49
5.4 Diagram representing main React Components. .................................. 50

6.1 Questionnaire: answers-1. ................................................................. 58
6.2 Questionnaire: answers-2. ................................................................. 58
6.3 Questionnaire: answers-3. ................................................................. 58
6.4 Questionnaire: answers-4. ................................................................. 59

C.1 Game starting screen. ........................................................................ 89
C.2 Not enough users joined to play screen. ............................................ 89
C.3 Game design sample. ........................................................................ 89
C.4 Leaderboard page. ............................................................................ 89
C.5 About page. ....................................................................................... 90
List of Tables

2.1 Characteristics of a crowdsourcing platform. ........................................ 12
2.2 Score design patterns for crowdsourcing types. .................................. 19

3.1 Crowdsourcing characteristics and the LdoD Archive. ...................... 28
3.2 Gamification characteristics and the LdoD Archive. ......................... 29

5.1 API Authentication Controller documentation. ................................. 41
5.2 API User Controller documentation. ............................................. 41
5.3 API Virtual Edition Controller documentation. ............................... 42
5.4 API Classification Game Controller documentation. ......................... 42
5.5 Classification Game Controller STOMP methods. .......................... 44

6.1 Fragments and its tags. .............................................................. 60
## Listings

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>WebSocketConfig.java</td>
<td>42</td>
</tr>
<tr>
<td>5.2</td>
<td>Usage example of a STOMP broker.</td>
<td>43</td>
</tr>
<tr>
<td>5.3</td>
<td>ScheduledTasks excerpt related with Game Runner.</td>
<td>45</td>
</tr>
<tr>
<td>5.4</td>
<td>Creating a scope called “thread” of the SimpleThreadScope class from Spring.</td>
<td>45</td>
</tr>
<tr>
<td>5.5</td>
<td>Registering the “thread” scope created above.</td>
<td>46</td>
</tr>
<tr>
<td>5.6</td>
<td>GameRunner code excerpt.</td>
<td>47</td>
</tr>
<tr>
<td>5.7</td>
<td>Usage example of react-stomp.</td>
<td>51</td>
</tr>
<tr>
<td>A.1</td>
<td>DML for the extended domain.</td>
<td>70</td>
</tr>
<tr>
<td>B.1</td>
<td>ClassificationGameController.java</td>
<td>73</td>
</tr>
<tr>
<td>B.2</td>
<td>ClassificationGame.java</td>
<td>79</td>
</tr>
<tr>
<td>B.3</td>
<td>GameRunner.java</td>
<td>84</td>
</tr>
</tbody>
</table>
## Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Program Interface</td>
</tr>
<tr>
<td>CAPTCHA</td>
<td>Completely Automated Public Turing test to tell Computers and Humans Apart</td>
</tr>
<tr>
<td>DH</td>
<td>Digital Humanities</td>
</tr>
<tr>
<td>DML</td>
<td>Domain Modelling Language</td>
</tr>
<tr>
<td>DTO</td>
<td>Data Transfer Object</td>
</tr>
<tr>
<td>GLAM</td>
<td>Galleries, Libraries, Archives and Museums</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>HTML</td>
<td>HyperText Markup Language</td>
</tr>
<tr>
<td>IST</td>
<td>Instituto Superior Técnico</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>JSP</td>
<td>JavaServer Pages</td>
</tr>
<tr>
<td>JS</td>
<td>JavaScript</td>
</tr>
<tr>
<td>JWT</td>
<td>JavaScript Object Notation Web Token</td>
</tr>
<tr>
<td>LdoD</td>
<td>Livro do Desassossego</td>
</tr>
<tr>
<td>MDA</td>
<td>Mechanics Dynamics Aesthetics</td>
</tr>
<tr>
<td>MVC</td>
<td>Model View Controller</td>
</tr>
<tr>
<td>ORM</td>
<td>Object-Relational Mapping</td>
</tr>
<tr>
<td>REST</td>
<td>Representational State Transfer</td>
</tr>
<tr>
<td>STOMP</td>
<td>Streaming Text Oriented Messaging Protocol</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>TEI</td>
<td>Text Encoding Initiative</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
<tr>
<td>UI</td>
<td>User Interface</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>
1 Introduction

Contents

1.1 Motivation ................................................. 2
1.2 Goals ....................................................... 3
1.3 Document Outline ......................................... 4
1.1 Motivation

The number one benefit of information technology is that it empowers people to do what they want to do. It lets people be creative. It lets people be productive. It lets people learn things they didn’t think they could learn before, and so in a sense it is all about potential.

Steve Ballmer

The above quote highlights the new day and age of our society: we can access, learn, engage, communicate, create and much more online. From an organizational perspective there is an opportunity to reap the benefits of that potential that exists in every Web user. Crowdsourcing is a way to achieve that, an example is Wikipedia, which is a collaborative online encyclopedia where users can read, create, extend and review information, and it’s all done by volunteers. The emergence of Web platforms and sites, resulted on the surge of crowdsourcing, making it both a business alternative as well as a useful option in the Galleries, Libraries, Archives and Museums (GLAM) industry.

Although the definition is very recent, 2006 [5], the notion exists from years ago, presumably the very first crowdsourcing project was The Longitude Prize designed by the British government in 1714, that sought a way to measure a ship’s longitudinal position and offered a prize to the best solution.

Besides the surge of crowdsourcing, another concept has seen rise in interest which is gamification. Despite the fact that the term was introduced in the 2000’s [6], only after 2010, there was a real growth in the use of the concept. In this thesis, we will also study the notion of gamification, through examples, and try to successfully implement a crowdsourcing activity integrated with gamification techniques within the context of a Digital Humanities project.

Fernando Pessoa is hailed as one of the finest Portuguese authors, best known for his countless poems, heteronyms and his magnum opus The Book of Disquiet (in portuguese, Livro do Desassossego (LdoD)). The book was published posthumously and is signed under Bernardo Soares, one of Pessoa’s heteronyms.

Pessoa died in 1935, leaving behind multiple manuscripts of his work in the form of fragments. These fragments revealed the existence of the LdoD, yet the book was found incomplete, without a particular order and with at least two heteronyms as authors of the work. After almost 50 years of research, study
and transcription of said fragments, the book was published in 1982 by Jacinto Prado Coelho. However due to the absence of a specific ordering for the texts, three more editions appeared, Teresa Sobral Cunha in 1990-1991, Richard Zenith in 1998 and Jerónimo Pizzaro in 2010, each one with different and diverse interpretations of the book.

These editions diverge on many things such as fragment organization, chronological order, heteronym attribution for the fragments and even spelling. This heterogeneity coupled with the absence of certainty about the author’s intentions generates a greater need for the reader to be able to read and understand each edition and then decide for himself what makes sense to him. As such, the LdoD Archive\(^1\) appears as an important tool enabling the construction of multiple reading paths while simultaneously promoting the work of the author. The Archive provides tools for reading and comparing editions; creating virtual editions and annotations; and sharing each edition with other users. After understanding the unique characteristics of the Book of Disquiet as well as its relation with the LdoD Archive, then the question arises \textbf{How can crowdsourcing and gamification techniques be used in order to enhance the collaboration in the context of the LdoD Archive?}

\subsection*{1.2 Goals}

Given the chaotic manner in which the Book of Disquiet originated and the cultural purpose of bringing together people around the work of Fernando Pessoa, one of the goals is to significantly improve the quality of reader’s digital collaboration and interaction with the archive. One aim is to enable users to have a pleasant interaction while fostering them to keep on working together. From a technical perspective, we want to find a way to give users tools that foster the interaction with the Archive and the collaboration in its context. So in order to discover these potential tools, in this thesis we will also thoroughly search for techniques and processes of crowdsourcing, and from that research we will propose an approach, well integrated in the archive context, that allows us to achieve our objective. Likewise, another objective is to ensure that volunteers in the crowdsourcing activity stay motivated and enjoy the experience, as such we will investigate the concept of gamification and try to include this onto our crowdsourcing process. Another goal is to integrate our solution with the Virtual Editions of the LdoD Archive, as such we must harness the existing features of virtual editions (explained in the 2.3) in order to develop a prototype that uses crowdsourcing and gamification.

Merging these two objective into a single main goal, it can be then formulated as follows: \textbf{Our goal is to carry out a comprehensive research of crowdsourcing techniques, processes, platforms, and other solutions as well as explore gamification techniques in line with the area of Digital Humanities, in order to be able to propose a suitable solution within the context of the LdoD Archive.} \\
\(^1\)https://ldod.uc.pt/
Archive, using its Virtual Editions which can then promote and enhance the interaction with the Archive, the collaboration of its online community and generate new crowdsourced artifacts.

1.3 Document Outline

The document is divided into seven chapters. In the next chapter (2), we introduce and uncover the architecture of the LdoD Archive, as well as the bibliographic research on crowdsourcing and gamification. Chapter 3 describes our problem, how it connects the Archive’s virtual editions with gamification and crowdsourcing, while also introducing what should our solution include. Afterwards, chapter 4 details the game, objectives and its design. Then, in chapter 5, the technical aspects of the solution are elucidated. Finally, in chapter 6 we present the results and at last in chapter 7 draw conclusions from our work and what could be done in the future.
Related Work

Contents

2.1 The LdoD Archive .................................................. 6
2.2 Architecture of the LdoD Archive ................................ 6
2.3 Virtual Editions in the LdoD Archive ............................. 9
2.4 Digital Humanities .................................................. 9
2.5 Crowdsourcing ....................................................... 10
2.6 Gamification .......................................................... 12
2.7 Crowdsourcing in the Digital Humanities ....................... 14
2.8 Gamification in Crowdsourcing .................................... 17
2.9 Gamification in Digital Humanities ............................... 19
2.10 A case study: Transcribe Bentham ............................... 20
2.11 ESP game: An analysis ............................................ 21
2.1 The LdoD Archive

The LdoD Archive is a collaborative digital archive based on the magnum opus by Fernando Pessoa, Book of Disquiet. The archive follows two main principles: representation and simulation. The history of writing and editing the Book can be seen as the representation principle. Additionally the possibility for users to read, edit and write in the digital medium, embodies the simulation aspect. The archive has multiple goals, it is noteworthy the goal to develop a virtual environment with multiple purposes, such as leisure reading, scholarly research and literary creation; enable the possibility of comparing the four main editions of the Book; to encourage different ways to read the Book of Disquiet and finally to promote the use of software tools that can be used to change the current analysis and interpretation predominantly based on printed literature.

In the following sections we expand on the architecture of the LdoD Archive (2.2) and on one of its functionalities, Virtual Editions, with great relevance and interest for this thesis (2.3).

2.2 Architecture of the LdoD Archive

We will now present the architecture behind the Archive according to the work of the authors in [1] and [7]. As we have seen, one of the goals is to have a platform for experts to study and compare different editions. While another objective includes a virtual archive available to anyone to experiment and produce different editions of the LdoD, as well as writing and extending original fragments. Furthermore, the platform should support the standard Text Encoding Initiative (TEI) \(^1\) encoding.

So firstly, the foundations of the features on the platform are based on four main functions: Reader, Editor, Book and Author. In Figure 2.1 we have a model of the LdoD Archive and these functions. The Reader-function essentially enables users to read fragments according to different authorial and editorial witnesses. The Editor-function as the name suggests, permits the view of the archive as a book, and allows the user the role of an editor, allowing text aggregation according to user defined criteria, including the addition of annotations and tags. The construction and rearrangement of information, allows the creating of a new edition, this is supported by the Book-function. Finally, users can extend the original LdoD by writing new text and serve as an author, this of course is the Author-function.

Likewise, there are three dimensions Genetic, Social and Virtual, that can be executed within each function. The representation of the text according to the autographs by Fernando Pessoa is embodied in the Genetic dimension. On the other hand, the perspective of the editors is represented in the Social dimension. Finally, the most relevant and important dimension to this thesis is the Virtual dimension. This dimension refers to the user’s reconstruction of the book, thus generating new virtual editions that can be shared, hence fostering the collaboration among users.

\(^1\)http://www.tei-c.org/index.xml
In figure 2.2 we have a model with the important entities regarding the virtual dimension. Firstly, we have at the root our single instance LdoD, which then contains a set of Fragments and FragInter, which represents interpretations of fragments. FragInter can be from an authorial source SourceInter (printed, typescript or manuscript), from an editorial origin ExpertEditionInter (these are enclosed and ordered in the context of an ExpertEdition instance) or from a VirtualEditionInter (which similarly to the expert source, are enclosed and ordered in the context of a VirtualEdition). A virtual interpretation is
constituted by a transcription (authorial or editorial) and a set of tags and annotations. Furthermore, the interpretation of a fragment has an associated \textit{Heteronym} which can be a virtual one, which represents an heteronym created by users.

The architecture behind the \textit{Archive} present in Figure 2.3 is composed by an Application Server; an Object Oriented Repository; a TEI Repository; a TEI Importer and Exporter and finally an Editor. With this architecture, the archive encoded as TEI is transformed into a set of objects through which a dynamic interaction becomes possible, in terms of editing.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{LdoD_Archive_Architecture.png}
\caption{LdoD Archive Architecture \cite{1}.}
\end{figure}

Furthermore, from a more concrete point of view, the \textit{Archive} has six features:

1. Reading: allows users to read the fragments according to each edition, and knowing where each fragment appears in the other editions. This feature is coupled with a recommendation system, which suggests the user what the next fragment to read given the set of fragments previously read and the following criteria: heteronym, date, text and taxonomy.

2. Documents: this section of the \textit{LdoD Archive} has important pieces of history, mainly witnesses which are elucidations of the fragments made up of interpretations and furthermore of printed texts by Pessoa, alongside some peculiar and interesting metadata. Besides this, it includes encoded fragments.

3. Editions: functionality that enables the viewing of different editions, such as the four main published editions, a virtual edition made by the team behind the \textit{LdoD Archive} and many other virtual editions that can be created by users of the platform which can make it public and available for others.

4. Search: includes two types of searches, a simple and an advanced one. The simple search uses the title as criterion, inquiring in the expert editions and witnesses. In turn, the advanced search
allows for the use of a rich set of possible combinations according to several criteria, such as edition, heteronym, manuscripts, taxonomies, text and date.

5. Virtual: one of the most interesting features, enabling the reader to create virtual editions and taxonomies, as well as sharing it with the community by making them public.

6. Writing: finally, this feature is not yet implemented, however its purpose will be to allow the writing of new texts that will be based on the fragments.

2.3 Virtual Editions in the LdoD Archive

One of the above mentioned features was the Virtual aspect, in the following section we expand on this concept, as it will be a key point in this thesis.

Firstly, as we have seen the Book of Disquiet is comprised by multiple fragments, which if rearranged in different orders (since it does not exist an authorial definitive order) can lead to different interpretations of the book, which as stated in 2.1 originated four main consensual editions among scholars. However, this absence of a natural and obligatory order for the fragments allows readers to have any interpretation and different reading experiences. From this, the LdoD Archive created the Virtual functionality allowing users to create their own editions of the book.

So, a virtual edition consists of a selection of fragments made by one or more users. This is a unique entity because it allows its virtual editors to choose the fragments they want to include, their ordering, and their annotation through notes and categories, and a virtual edition can be public or private.

The interesting part of virtual editions, besides its uniqueness, is the collaboration aspect, because a virtual edition can have multiple editors, i.e, users select fragments, categorize and annotate them.

This feature is available at https://ldod.uc.pt/virtualeditions and the only requirement to create and join virtual editions is to be a registered user of the platform.

2.4 Digital Humanities

Digital Humanities, is a field that combines the traditional humanities with the digital environment. The genesis of the Digital Humanities, as David Berry introduces in [8], is very humble, in way that initially it wasn’t even a field, simply serving as support for the research of other fields, and being originally called “humanities computing”. As it became more relevant, a switch to Digital Humanities occurred and meant to signal that the field had emerged from the low-prestige status of a support service into a genuinely intellectual endeavour with its own professional practices, rigorous standards, and exciting theoretical explorations [9]. A good definition is provided by Presner in [10] stating that it’s an umbrella term for a
wide array of practices for creating, applying, interpreting, interrogating, and hacking both new and old information technologies.

From these definitions, we can draw a parallel and understand that the LdoD Archive fits perfectly in this area of study. Given this, we must keep in mind the scope of Digital Humanities (DH) when studying and researching for crowdsourcing techniques (2.6) as well as gamification processes (2.8.1).

2.5 Crowdsourcing

With the development of the Web, new areas of interest have emerged, one of them being crowdsourcing. The term crowdsourcing first appeared in an issue of Wired magazine [5] and was later defined as: the act of a company or institution taking a function once performed by employees and outsourcing it to an undefined (and generally large) network of people in the form of an open call. This definition also states that The crucial prerequisite is the use of the open call [11].

For now let’s put the definition on hold, and consider some other concepts related with crowdsourcing such as open and user innovation [12]; cognitive diversity and wisdom of crowds [13]; collective intelligence and finally cognitive surplus [14]. Regarding the cognitive surplus, Clay Shirky explains that the way a human uses his/her free time can be a seen as a shared global resource and it can be advantageous to explore it [14]. This is one of the driving forces that lead to the emergence of crowdsourcing, because it enabled people to have the desire and time to even consider to engage in a crowdsourcing process. Similarly, James Surowiecki analyzes in his book The Wisdom of Crowds how a group of people can sometimes, with certain conditions, outsmart any individual of the group. In order for that to happen, he states that a crowd needs to satisfy three conditions: be diverse; the members must be independent; and it must have a “particular kind of decentralization”. Beside this, Eric von Hippel writes that improvements in computer software and hardware are making it possible to develop increasingly capable and steadily cheaper tools for innovation that require less skill and training to use [12]. To conclude, all of these concepts are intrinsically related to crowdsourcing, in the sense that they help to explain the emergence and necessity of crowdsourcing.

In the paper Towards and Integrated Crowdsourcing definition [15] the authors discovered that forty definitions were published between 2006 and 2011. The authors then synthesized that research into a single definition, where we highlight the following characteristics: participative online activity proposed via a flexible open call to a group of individuals of varying knowledge, heterogeneity, and number, and the undertaking of the task (. . . ) always entails mutual benefit. This definition is our starting point, because it synthesizes many other definitions into one, however we will try to mold and adapt this definition to our specific purpose and context.
2.5.1 Dissecting crowdsourcing through four pillars

From the work of [16] we can achieve a better understanding of crowdsourcing. The authors of the paper, explain crowdsourcing through four pillars:

- The **crowd**: the participants of the crowdsourcing activity;
- The **crowdsourcer**: the entity looking to fulfill a task through the power of the crowd;
- The **crowdsourcing task**: the activity that is asked to be completed by the crowd;
- The **crowdsourcing platform**: the system where the task is to be completed.

The authors then identified the main characteristics about each pillar. Five distinct features about the crowd were identified:

1. **Diversity**: enlistment of different individuals, it can be spatial (different backgrounds and locations), gender, age and even expertise diversity;
2. **Unknownness**: the crowd does not know the crowdsourcer nor the other members;
3. **Largeness**: ensure that the crowd is large enough in order to achieve the task;
4. **Undefinedness**: the determination of the crowd has no imposed conditions;
5. **Suitability**: the fit of a crowd for performing crowdsourcing activities.

Regarding the crowdsourcer, four main features were established:

1. **Incentive provision**: incentives (financial, social or entertainment) as way of extrinsic motivation;
2. **Open call**: crowdsourcing is generally open to the public, similar to undefinedness, but from the crowdsourcer perspective;
3. **Ethicality provision**: ensure some ethical standards during the crowdsourcing activity, such as Opt-out procedure: crowd can stop the activity; Feedback: provide feedback to the crowd; No harm: ensure that crowd will not be harmed during the activity;
4. **Privacy provision**: private information of the participants should not be disclosed.

The task itself has eight characteristics:

1. **Traditional operation**: the execution of the task if it was not crowdsourced;
2. **Outsourcing**: usually a task that would otherwise be outsourced;
3. **Modularity**: modularity means to split something into different parts and components, in this case a task due to being too complex is divided in micro-tasks;

4. **Complexity**: complexity of the task, (complexity ≠ modularity, although being closely related), a task can be complex and atomic;

5. **Solvability**: a task too complex for machines but usually simple enough for humans;

6. **Automation**: task either difficult or expensive to automate;

7. **User-driven**: a task powered by humans (usually solving; co-creation and innovation);

8. **Contribution type**: can be individual or collaborative.

Finally regarding the platform, it can be said that it has four characteristics, two related with the interactions on the platform and two regarding the facilities provided by the platform. Those characteristics are presented in Table 2.1.

<table>
<thead>
<tr>
<th>Crowd-related interactions</th>
<th>Interactions between the crowd and the platform. Examples: enrollment, authentication, assignment, submission, supervision, feedback.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowdsourcer-related interactions</td>
<td>Interactions between the crowdsourcer and the platform. Examples: similar to crowd, and time/price negotiation, verification.</td>
</tr>
<tr>
<td>Task-related facilities</td>
<td>Facilities about the crowdsourced task. Examples: aggregation, hiding results, history, threshold.</td>
</tr>
<tr>
<td>Platform-related facilities</td>
<td>Facilities about the platform itself. Examples: online environment, managing, payment, interface feasible and attractive.</td>
</tr>
</tbody>
</table>

**Table 2.1**: Characteristics of a crowdsourcing platform.

### 2.6 Gamification

Another considerable component of our research is gamification. As we will see, this phenomenon has risen as a key factor to motivate user across multiple contexts. For instance, nowadays many applications have turned to gamification. “Todoist” is an example: it’s an application for personal and professional use, where users manage tasks among other things. However the key notion is that for each task completed, the user is awarded points, introducing a game element and motivating users to keep on finishing the tasks. Another interesting example is “Chore Wars”, which, as the name suggests, is an application that gamifies the old and boring house chores. Gamification can then be defined as the use of game design elements in non-game contexts [17]. The term is relatively recent, having appeared
in 2008, but only gained traction in 2010. In [17], Deterding provides a definition for each term. From there we synthesize each term as follows:

- a game follows rules and has defined goals;
- elements are characteristics encountered in most games and that have a significant impact in the gameplay;
- the design is characterized by varying levels of abstraction and five designs were identified (ordered from concrete to abstract):
  - interface design patterns, for instance badges;
  - design patterns and mechanics, time constraint is an example;
  - design principles and heuristics, for example clear goals;
  - models, a fantasy model for instance;
  - design methods, such as playtesting.
- non-game contexts, which is not clearly limited by the author, because there are no specific usage contexts, however it excludes the use of game design elements as part of designing a game, since that would simply be game design.

The relevance of gamification to this thesis is due to the fact that gamification presents itself as a potential solution in order to create, support and boost the interaction between users and the archive, and even among users themselves. In order to expand our knowledge about games and gamification, we now focus on a formal framework Mechanics Dynamics Aesthetics (MDA), [18] which yields a convenient approach to understanding games and their design. The MDA framework endeavors to connect the game design, development, feedback, and technical game research. Firstly it structures the model into two perspectives: the player and the designer, in which one consumes games and the other creates games, respectively. The consumption process is split into three categories rules, system and fun. Equivalently, there are three counterpart members from the designer’s perspective of the game, they are mechanics, dynamics and aesthetics. Mechanics are specific segments of the game, at the level of algorithms and data, for instance consider a shooter game, the mechanics include guns, munition and spawn areas. Dynamics portrays the mechanics run-time execution based on the input and outputs of the players over time, once again considering the earlier example we can say that shooting and spawning are some of the dynamics produced. Finally, aesthetics depicts the sentimental reactions evoked in the player when engaging with a game, and can be categorized into sensation, fantasy, narrative, challenge, fellowship, discovery, expression, and submission. Considering yet again our shooter game as an example, our categories are mainly challenge, sensation and fellowship, specially if the shooter is
team-based. When designing our gamified solution, this framework can be very helpful in order to help us design a complete and immersive experience.

2.7 Crowdsourcing in the Digital Humanities

As we have seen, Crowdsourcing constitutes an activity that is outsourced, this can be specially useful within the concept of Digital Humanities, specifically within GLAM. Nonetheless crowdsourcing has its origins in the business world and we have to redefine it for the area of cultural heritage, explains Owens in [19]. Firstly we will adapt and further understand crowdsourcing in the Digital Humanities, then present examples and draw some conclusions.

Contrary to a traditional business oriented crowdsourcing, GLAM have neither the financial incentive resources nor seek profit, their purpose is however to share knowledge and culture with the general public. Owens [19] highlights important notions such as: the work is not labour but an invitation to meaningful work; the crowd is replaced by engaged enthusiast volunteers; volunteers are amateurs, not in a derogatory way, but in the true sense of the word, which describes someone that dedicates to study something in a non-professional manner. Finally Owens synthesizes crowdsourcing as a complex interplay between understanding the potentials for human computation, adopting tools and software as scaffolding to aid this process, and understanding human motivation.

Subsequently intriguing examples of crowdsourcing in Digital Humanities have been found such as:

- the Australian Newspaper Digitisation Program\(^2\). A project released to users in August of 2008, which asked citizens help in order to digitalize thousands of articles. The project had, until November 2009 approximately 6 000 volunteers and 7 million lines of corrected text over 318 000 articles;

- the United States Holocaust Museum’s “Remember Me” project\(^3\). In this project, the museum seeks the help of the crowd in order to identify children in old photos taken during the War, and through that facilitate the reconnection between survivors, families and other individuals;

- Galaxy Zoo\(^4\) which is an online platform to classify galaxies involving individuals as well as universities. The project is a part of Zooniverse\(^5\) which is a platform that aggregates other citizen science projects. The Galaxy Zoo \(^1\) project was released in July of 2007, gathered over 150 000 volunteers and 50 million images of galaxies had been classified and it was retired in 2009. However, due to the project success, Galaxy Zoo has evolved and new projects have been developed

\(^2\)www.nla.gov.au/content/newspaper-digitisation-program
\(^3\)https://rememberme.ushmm.org/
\(^4\)www.galaxyzoo.org
\(^5\)www.zooniverse.org
\(^1\)http://zoo1.galaxyzoo.org

14
and retired. As of 2017 the Galaxy Zoo platform has had 15 projects, which is testament to the success of the platform;

• Letters of 1916\footnote{http://letters1916.maynoothuniversity.ie}, a project that seeks to have a digital collection of letters from the Easter Rising in Ireland. The project began in 2013, based on a digital collection that include letters from institutions and private collections. The letters range from many topics such literature, art, the Easter Rising, the Great War, politics and much more. On March of 2016, it was released the access to the database, allowing users to make transcriptions and explore the collection, through date, place, topics and being able to see images and original transcripts. The project currently has over 1300 transcribers.

Schreibman in \cite{20} gives an introduction to Crowdsourcing in the Digital Humanities, presenting these projects and others, stating that the relationship between the projects is that GLAM institutions, who are custodians of such historical material, often partner with University researchers who have an interest in using digital techniques to answer their Humanities or Heritage based research question, this conclusion of the author is quite relevant, because it presents parallels to the genesis of the LdoD Archive.

Schreibman then references \cite{21} a paper written by Rose Holley, a manager in the Australia Digitisation Program\footnote{2}. Rose brings insightful tips such as a clear goal, report regularly on progress; system should be easy and fun and volunteers should be acknowledged, rewarded and supported by the project team.

Given all this, a more fitting definition could be the one by Ridge in \cite{22}, that states crowdsourcing as form of engagement with cultural heritage that contributes towards a shared, significant goal (…) by asking the public to undertake tasks that cannot be done automatically, in an environment where the tasks provide inherent rewards for participation.

Besides the surge of crowdsourcing projects in Digital Humanities, there’s also been an increasing interest in the research area: in a 2012 study \cite{23}, 54 academic publications and 51 individual projects, activities or websites related with humanities were identified. The authors from the study, also identified four main factors that define crowdsourcing within humanities research. These are:

• a research question related with humanities;

• the possibility for an online gathering to join the project;

• a perceptible task which is fragmented into an achievable work process;

• the setting up of a scalable activity which can be undertaken with different levels of participation.

Another important paper is Crowdsourcing in the Cultural Heritage Domain: Opportunities and Challenges \cite{2} that presents, using the Digital Content Life Cycle, how can we achieve a more open, con-
nected and smart cultural heritage, as such the paper studies different types of crowdsourcing projects. Firstly, the methodology was defined as:

1. categorize the projects within the GLAM scope;
2. identify correlations between the Digital Content Life Cycle and the GLAM area;
3. detect opportunities and possible challenges.

In order to find the classifications for crowdsourcing projects, the authors have studied other papers that have already done that research, and from that the final categories were as follows:

- Correction and Transcription Tasks;
- Contextualisation;
- Classification;
- Co-curation;
- Crowdfunding.

There are many models that help understand the challenges in the cultural heritage organizations, however this paper [2] uses the Digital Content Life Cycle model from the National Library of New Zealand. As we can see in Figure 2.4, five of the six crowdsourcing types are associated with the stages of the model, the sixth one (crowdfunding), can be associated in any stage. The model, although being cyclical, it doesn’t necessarily need to be like that, some activities can be done along many stages. This model is mainly a simplified view of the activities in the GLAM sector and can be a starting point to plan a crowdsourcing activity. Their website contains helpful tips in order to achieve the best possible results.

Finally, through all this, the paper identifies opportunities and challenges, the two most important being an adequate instruction of the user and the creation of a faithful foundation base of user’s in a manner that achieves a sensible level of quality. The first challenge, brings the opportunity of using motivational factors in order to achieve the faithful foundation of users. The authors using existing crowdsourcing projects have identified some factors that are important to foster such as connectedness and membership; sharing and generosity; altruism, fun and competition. The other main challenge is quality assurance in relation to which the authors highlight that About 90% of the online users only consume content and from the 10% left only 1% actively and consistently contribute the majority of the user-generated content and 95% of this content is either spam or malware. In addition, the GLAM sector is not allowed to have misinformation: imagine going to a library, or a museum and obtaining wrong information, so the standard is higher in this area. Some guidelines are drawn regarding the quality, namely to enforce community norms and have filter mechanisms in place.

2.8 Gamification in Crowdsourcing

Recalling one of the topics introduced in the previous section, we noticed that user’s motivation is in fact a basilar point in crowdsourcing, in this way we will study gamification in crowdsourcing as a tool to increase and maintain the motivation of the users. As such we will use the work of [3] as reference, where the authors conduct a review of gamification in crowdsourcing. Firstly the authors, present the work of [24], which they categorize into four different types of crowdsourcing activities. Each one having specific characteristics, from two dimensions, one that focuses on where the value of the contribution is originated and the other on how it is valued different contributions. They can be classified as:

- Emergent (the whole of contributions make the value) or Non-emergent (each individual contribution has a value);
- Homogeneous (each contribution is equally valued) or Heterogeneous (each contribution can have a different value).

Firstly, we have crowdsolving, which can be characterized by using the variety of the group to locate a huger number of heterogeneous answers for a given issue. It is regularly utilized for extremely complex issues or if no pre-perceptible solution exists.

Then crowdcreation, which is to create artifacts based on a diversity of heterogeneous contributions. Youtube which represents an enormous set of different user-generated content and Wikipedia which combines knowledge from multiple contributions are examples of this archetype.
Crowdrating, is the third archetype, and can be described as frameworks which ordinarily endeavor to use the *wisdom of crowds* to perform assessments or forecasts, in this instance through votes.

Finally, crowdprocessing, which includes methodologies that depend on the group to perform extensive amounts of homogeneous assignments. Similar contributions are a quality trait for the legitimacy of the work. The value is non-emergent.

### 2.8.1 Gamification Recommendations

In the paper [3], the authors expanded on how gamification works and conducted a literature review and presented their results, including some recommendations.

- Points/Scores: Most of the systems studied had a metric as core element in order to award users.

- Ranking/Leaderboards: Rankings have shown to be effective in motivating users.

- Level systems: Another feature that appeared to be effective, specifically social achievements.

- Manifold gamification approaches: A mix of gamification designs can be useful, however it must be noted that not all motivational affordances produce necessarily better results, so its important to create a design focused on user needs.

The recommendations for each archetype are summarized in Table 2.2.
<table>
<thead>
<tr>
<th>Crowdprocessing</th>
<th>Should award the completion of tasks, usually through level systems and additionally leaderboards to accomplish focused engagement, must be used with caution.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowdrating</td>
<td>Focus on quality and quantity. Scoring, leaderboards and time pressure can be used to create competition-based setting, forcing the think and act like the community notion.</td>
</tr>
<tr>
<td>Crowdsolving</td>
<td>Explicit gamification rewards before a crowdsolving task phase can increase the quality of CS work and engagement. An open chance to achieve higher rewards depending on quality of work can also increase engagement.</td>
</tr>
<tr>
<td>Crowdcreating</td>
<td>Gamification should be used to motivate users towards diverse and creative contributions.</td>
</tr>
</tbody>
</table>

Table 2.2: Score design patterns for crowdsourcing types.

2.9 Gamification in Digital Humanities

As we’ve seen there has been a rise in digital technologies, and the GLAM industry has put an effort in harnessing those technologies. This is due to the fact that GLAM's are an interesting case in which non-profit organizations have a semi-business model, because those organizations still need to justify their existence through number of visitors for example. Therefore gamification began to be perceived by GLAM's as a tool, which through a competition environment can help in the promotion of the industry; increase the use of collections and build a more interactive relation with citizens. Interesting examples can be found in the Gamification of libraries, the [25] paper presents some of those cases.

For instance, the Library of the University of Huddersfield in the UK, gamified the library experience by developing Lemontree, a social game where users gain points and badges by engaging with the library such as visiting, checking a book and others. Also those badges, could then be shared and displayed on Facebook and Twitter. The purpose of the game was to mainly increase the visitor interaction and the reports showed that it was successful.

Another impressive example was the North Carolina State University Libraries which gamified the process of library instruction. Traditionally, the process consisted of simply showing the library, the website as well as other resources, and how the access was made. However these Libraries, created a mobile app as tool for a scavenger hunt around the library, encouraging students to interact with the staff, the collection of book's and more. The scavenger consisted in students splitting into teams, each of which had to answer fifteen questions in twenty five minutes while exploring the library. After the time was up, the team would return to the classroom and find the correct answers, and the team that won would have some prizes. The project was a success, having reached more than one thousand and six hundred students over ninety scavenger hunts. The feedback also supports this, both faculty, students, and staff responded positively to the scavengers.
2.10 A case study: Transcribe Bentham

We will now focus our attention on the work from the paper “Many Hands Make Light Work. Many Hands Together Make Merry Work”: Transcribe Bentham and Crowdsourcing Manuscript Collections which is part of the book Crowdsourcing our Cultural Heritage [26]. Firstly, some context for this project. Jeremy Bentham was an English philosopher and is known from his work in Economics, Law reform and mainly is regarded as the founder of Utilitarianism. Bentham was also known for pushing for innovation and development, as exemplified by his wish to have his remains to be publicly dissected, setting the way for others to do the same and keep science evolving. Bentham’s work contains important and interesting knowledge, thus the collection of his manuscripts, held by University College London (UCL), has been released online via a web platform and made available to scholars, students and the public in general, in order to promote knowledge.

From this, the Transcribe Bentham project was born, with three main goals: allow the transcription and publication of Betham’s papers in the UCL website, where they become accessible and searchable; enable volunteers to engage with digital humanities and be part of something big, and finally the project served as an experiment, in order to find out if a complex crowdsourcing project like this, and given the lack of the volunteer’s training, could be successful and worth it.

The first phase of the project was the design stage where the system was conceived. The main component is the “Transcription Desk” which is a personalized installation of “MediaWiki”. “MediaWiki” is a free and open-source software dedicated to wiki management, and it’s currently used for instance in Wikipedia. The usage of “MediaWiki” in Wikipedia means that the interface is easy and familiar to many. Alongside this some factors like the fact that the software is stable and has a good documentation, contributed to the decision of using it. Thus the “Transcription Desk” runs on “MediaWiki” and aggregates user management; the transcription platform; the current progress of the project; the leaderboard and news regarding the project. Another important part of the platform is the encoding of the manuscripts, which follows the now standard TEI that defines a Extensible Markup Language (XML) format. Furthermore, due to the concerns of volunteers not having notions of TEI or markup, two extensions were developed (JBZV and JBTEIToolbar), that together support the Transcription Toolbar, that makes easier for volunteers to mark paragraphs, headings, paragraphs, additions, deletions and much more, in a forthright way, without a need to have background knowledge of mark-up.

The project was launched in September of 2010, for a six-month period of testing. However, the project only had some success after New York Times wrote an article, in December, and that leveraged the number of volunteers and transcriptions made. In the first three months, it had 350 users and 439 manuscripts transcribed (fully or partially), and at the end of the six months, 1,222 users and 1,009

---

9 http://www.transcribe-bentham.da.ulcc.ac.uk/td/Transcribe_Bentham
10 www.mediawiki.org/wiki/MediaWiki
manuscripts transcribed (559 of them being complete).

Thus, the project entered a second stage, Consolidation, spanning from early March of 2011 to late September of 2012. This stage is characterized by the project having a reduced budget, resulting in cutting full-time staff and only having money for two-days per week of staff time, and the rest of the money covered web storage costs. Despite this, the project kept on growing, with more users and manuscripts transcribed, which culminated in the project being mentioned in a Sunday Times article and even receiving the Award of Distinction in the Digital Communities category of the 2011 Prix Ars Electronica. However new fears emerged, one being the stagnation of the project. This concern was real and was addressed by a survey made to the volunteers in order to find out their motivations. The key takeaways were that volunteers felt that time was becoming a limiting factor in participating, due to the difficulty of understanding the calligraphy of Bentham and the use of TEI mark-up also complicated the task.

A new and current stage is presented, Expansion (October 2012 - onwards), this phase is characterized by new funding and an enhanced and updated Transcription Desk, that tried to meet the needs reflect by the volunteers in the aforementioned survey. The new and improved Transcription Desk included new features as an image viewer that allowed transcribers to rotate, maximize and minimize the manuscript. Besides this, a preview of the encoded transcript was added as well as a “Changes” tab, that enables the user to see the changes made in the transcript. These two features were made with the goal of diminishing possible errors and being more time efficient (as opposed to having to save and check the final result every time). The changes seemed to have worked, and the project kept on growing, at this point having 2,934 users and 5,799 manuscripts transcribed (which were 95% complete).

2.11 ESP game: An analysis

The ESP game\textsuperscript{11} \cite{4} was a game developed in order to produce meaningful and relevant metadata information. This is a prime example of an idea that harnessed the potential of the crowd allied with the gamification of an otherwise boring process. As such it is relevant to this thesis to try replicate some of the principles applied as well as avoid the deficiencies and drawbacks of its implementation.

2.11.1 Motivation

The motivation for the game was the issue regarding the labeling of images on the Web. There is a high level of importance on having rigorous imaging metadata because it is useful for site accessibility (consider applications for visually impaired people); blocking of content not appropriate (graphic images

containing pornography, violence and others); overall good search imaging and also for computer vision research (large databases of labelled images used in machine learning).

This categorization had two main ways of being made, computer computations, which have shortcomings such as relying on images descriptions resulting in inaccurate results or manual labelling which is a tedious and costly task.

Therefore, the goal for the creators of this game was to create a way to address the metadata issue while producing meaningfully results in a way that people had fun and were entertained in the process.

2.11.2 Description

The game consists of two random online players from the pool of users in the game and it is made to be played in pairs only.

The flow of the game is very straightforward. Both users have the same image showed simultaneously to them and their goal is to “agree on an image”, this consists in during a limited time users must insert labels until each has submitted the same string and then they move on to the next image. However, the participants can pass or opt out of challenging images, in that case a message will appear on the screen of the other partner and a image will only be passed if both have agreed on passing. The users do not know one another, are not able to communicate with nor see the guesses of its counterpart.

| Player 1 guesses: purse | Player 2 guesses: handbag |
| Player 1 guesses: bag | Player 2 guesses: purse |
| Player 1 guesses: brown | Success! Agreement on “purse” |

Figure 2.6: Example of players agreeing on a guess [4].

Agreeing on an image results on points, large bonus points are also attributed if they agree on 15 images. The interface also shows the number of images agreed upon. User's decisions are reinforced and they are pushed to keep playing due to point attribution.

Since the participants cannot communicate the only strategy to follow is to type an obvious word related to the image, specially because the system asks players to “think like each other” not to describe
Another concept introduced in the game is Taboo Words, which as the name suggests means that these words that cannot be submitted (equally for singular, plurals or phrases containing these words). Taboo Words are related to the image, making the game more challenging. The game provides Taboo Words, the first time an image is used in a game it has no taboo word. However the next time the image is reused, the game will have assigned a taboo and so fourth (maximum of six distinct words). The motive is to generate as different word as possible and avoid generic labels, resulting in more specific labels created.

An important note however is the concept of good label threshold, which means that a word only becomes taboo if it meets a threshold, which may be set low as, a single pair agrees on a label, or very high, as when forty pairs agree on a label.

Additionally, the system also decides when a image is “retired”. Images are used throughout the game various times, as such the system must decide when one image should no be used again. This decision is made based upon the fact of having acquired an extensive number of taboo words, which can result of course in difficult guesses and getting passed by the participants. Furthermore, an image that was previously “retired” can return due to changes. For instance, an image of Lance Armstrong years ago could be something like “phenomenon” where nowadays it could be “cheater”.

### 2.11.3 Implementation

In [4], the game was implemented as Java applet that pairs up two random players, 15 images to categorize, including the corresponding taboo words and compare the strings submitted (at the time, maximum allowed of 13 characters).

Some game characteristics also include a spelling dictionary in order to avoid typos and a pre-recorded game play, meaning that one person could play with a bot. This bot is just a set of actions that had been done by another person in another game session (with two persons) and were recorded.

In order to prevent cheating, although unlikely due to the difficulty of creating an agreement strategy with the other partner, the system only starts every 30 seconds to avoid two friends connecting at the same time and try to meet each other. Furthermore, the Internet Protocol (IP) addresses are checked and must be different. Another concern is massive global agreement, for instance everyone starts to submit the character ‘a’ for every image, the system uses the pre-recorded play which will discourage this strategy. The strategy is detectable easily if the agreement time starts to be quicker than the average. Taboo words is also a different mechanism to prevent this, meaning that a word could become taboo for the entire session. All of this together with label threshold make it tough to cheat.

A suggested feature is theme rooms, which can generate more specific labels, since this way people can choose rooms that focus on specific types of images. For instance, an art room could generate
more specific labels like the author or title of the art in question, while a general room could have more
generic labels like “painting”. Similarly, inappropriate content (e.g. pornography) could appear in the
general room, however a children’s version could resolve this in which case the images in this version
would have to have been considered safe already.

2.11.4 ESP Game Evolution and Present

In 2006 the ESP game was purchased by Google, with the purpose of applying it to their own online
image results, renamed Google Image Labeler. Google’s own version was online between 2006 and
2011, and was relaunched again in 2016.

During that time period of 2006 through 2011 the game was mainly similar to the one conceived by
Luis von Ahn, however over time some small changes started to occur.

In the early months of 2006, players could see the other players guess by hovering over the images,
this lead to abuse as result of players now starting to submit certain words that they had learned where
usually guessed, resulting in higher scores and rigging the system. This was later fixed, and only at the
end of the game the guesses were available for checking.

In 2007, users started to notice when they were playing with a “bot” and, so, the game suffered new
changes in order to avoid this.

\[\text{https://crowdsource.google.com/imagelabeler}\]
Some other big changes involved changing the timer and the scored attributed, rewarding more specific words. The game suffered more changes including User Interface (UI), until 2011 when the game was discontinued.

Lastly, the game was relaunched with the same name, nonetheless it does not resemble any of the game’s origins. Nowadays it is mostly a classification service, where the user is presented with a category, selects one and can start playing. From there a set of images are presented and the player must indicate if the image belongs to that category.

An educated guess is that beyond being availed for improving search imaging results it is also an utility service for Google’s image **Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA)**, which are currently commonly used in most authentication systems to avoid “bots” spamming user credentials.

![Categories](image1.png) ![The game itself](image2.png) ![Achievements](image3.png)

**Figure 2.8:** Current Google Image Labeler.

---

13https://www.google.com/recaptcha/intro/v3beta.html
## Problem Statement

### Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Virtual Editions</td>
<td>27</td>
</tr>
<tr>
<td>3.2 Crowdsourcing and the <em>LdoD Archive</em></td>
<td>27</td>
</tr>
<tr>
<td>3.3 Gamification and the <em>LdoD Archive</em></td>
<td>29</td>
</tr>
<tr>
<td>3.4 The <em>LdoD Archive</em> and the ESP game</td>
<td>29</td>
</tr>
</tbody>
</table>
3.1 Virtual Editions

As we have stated in section 1.2, our intention is to ultimately promote and enhance collaboration in the LdoD Archive. In section 2.3, we presented and expanded the definition of a Virtual Edition, in this section we identify the main characteristics of interest from this existing functionality and draw a parallel with our problem and what should our solution focus on.

Our focus on the Virtual Edition aspect of the Archive is to use it as a tool to harness the potential of crowdsourcing and gamification, since it allows the collaboration aspect of users in virtual editions.

Let us consider a scenario in which a virtual edition could be useful and interesting. Consider a classroom, where students have a class virtual edition in which they study the work of Fernando Pessoa in a different and stimulating way, while the teacher has the control as a manager and the students are the editors.

For instance, their task could be to categorize the edition freely (and explain to the teacher their choices) or according to a vocabulary pre-defined by the teacher herself/himself.

This scenario is a starting point for this thesis, because this would be even more interesting for the students and the teacher, if it could be done in a interactive and real-time scenario, like in a online multiplayer game.

From this, our solution is to create a game that incorporates Virtual Editions, its ability to collaborate with user with the capability to classify fragments, in a way that we can construct multiple crowdsourced virtual editions that were developed in a gamified environment.

Regarding this game, it is important to understand that it must follow and comply with the rules of a virtual edition. For instance, for private virtual editions of course that only members of the edition can play games. On the other hand, if the virtual edition is public the game can be opened to entire universe of registered players or just the members of the virtual edition, and this is defined in the configuration of the virtual edition. The game simply follows that, it should not revoke configurations of the virtual edition.

Most importantly, for any game to occur the virtual edition associated must have an open vocabulary, this means that it must be available to be classified with any tags instead of a pre-defined set of tags (closed vocabulary).

3.2 Crowdsourcing and the LdoD Archive

Currently, the Archive already allows some crowdsourcing features, nevertheless in a very narrow and specific form of crowdsourcing.

The process in which a user creates a virtual edition, makes it public and enables the other users to participate in their edition by annotating and tagging, is already a a crowdsourced virtual edition.
However, this “crowdsourcing” project is somewhat constrained. The crowdsourcer (user that creates a virtual edition) does not have a way to make the open call in the platform, nor does he or she have a proper mechanism to specify with greater detail the goal and meaning of the task at the hand, nor can the crowdsourcer provide feedback to the crowd nor the other way around. So our focus is to provide tools that can extend this.

From this we can already identify three main issues: the absence of an open call; a dedicated platform (although the Archive allows crowdsourcing, it was not made directly and specifically for crowdsourcing) and a way to show appreciation and recognise the effort of the participants of the crowd.

In summary, in the 3.1 we introduce the six main points identified in a crowdsourcing project and relate them to the Archive as a way to understand what could be improved.

<table>
<thead>
<tr>
<th>Crowd</th>
<th>Our crowd can include experts (editors and staff for instance); students (Fernando Pessoa is an important author in the Portuguese learning system); Pessoa enthusiasts (public in general who appreciate the work of the author). From this, our crowd seems diverse, large and suitable to the project.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crowdsourcer</td>
<td>LdoD Archive editor (obviously the organization); virtual edition editor (a creator of a virtual edition can crowdsource, by allowing the public to annotate and tag his virtual edition) or for instance a teacher who seeks to use this as an education tool in his/her class.</td>
</tr>
<tr>
<td>Task</td>
<td>Currently some of the available tasks are tagging, annotating and creating virtual editions. These tasks are already significantly modular (tagging and annotating can even be considered micro tasks) and present some complexity (specially if the tasks are applied to an edition and not a fragment, such as rearranging the edition), and are all user-driven.</td>
</tr>
<tr>
<td>Platform</td>
<td>The platform already has available facilities such as user sign up; authentication; reading; editing and user management.</td>
</tr>
<tr>
<td>Crowdsourcer desirable features</td>
<td>It seems important that our project should have an ethicality and privacy provision. Besides that an open call should be made to the crowd. An incentive provision should also be put in place, however this incentive will rely on social and entertainment factors, powered by gamification in the project.</td>
</tr>
<tr>
<td>Platform desirable features</td>
<td>In our current platform we already have some facilities and interactions available, however some desirable features include: a new platform should be developed to support the game itself and other gamification features (score, ranking system and calculations); a way for users to provide feedback on the task and project; management of the crowdsourcing by having features such as a history (when was the game created, by who and in which date; which users contributed and with what suggestions and many other logging information).</td>
</tr>
</tbody>
</table>

Table 3.1: Crowdsourcing characteristics and the LdoD Archive.

From this our main target crowd are students (more enthusiastic and receptive to games, because it fits in the currents program of secondary schools); the crowdsourcer is the teacher; the task tagging; the incentives are game related and should serve as motivation point; and finally the new platform should have all of the necessary tools in order to allow the game to take take place, such as login; a server
connection that fetches data; the ability to communicate in real-time and the synchronization of games.

### 3.3 Gamification and the LdoD Archive

The other perspective in which we focus, is the gamification of the Archive. As expected, the Archive does not currently support game related characteristics nor a game in itself. However it does have some specific characteristics like a web page that act as a profile for the user, showing the virtual editions of which the user is a member and the fragments to which the user contributed with tagging and annotations.

With the existing characteristics of the platform and through MDA, our aim is to create a game to enhance the crowdsource of virtual editions. In the table 3.2, we present a summary of gamification characteristics and the Archive.

<table>
<thead>
<tr>
<th>Archetype</th>
<th>Given the creative and collaborating environment of the Archive, crowd-creating should be our reference model, where the contributions are emergent and heterogeneous. For instance, the game could consist of developing a new edition where users tag the edition as a group.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motivation and design</td>
<td>Our project must try to motivate users in a way to promote diverse and creative contributions, as such our design must include elements like a ranking and points. This way we can reward creative contributions and show public appreciation.</td>
</tr>
<tr>
<td>MDA to the Archive</td>
<td>In order to implement some game elements, the MDA framework can be helpful to rationalize some ideas to implement. Firstly, our mechanics could include a ranking system based on points given to users after completing interactions with the platform, and of course the rules that determine the points assignment are also a mechanic. The dynamics can be interactions with the platform and with other users such as: reading; tagging; commenting on existing editions, providing feedback, among others. Finally, the aesthetics part: as we have seen there are 8 categories, and the most adequate category to our goals is the fellowship which views the game as a social framework, which ultimately is our main ambition, to generate interaction in the community.</td>
</tr>
</tbody>
</table>

Table 3.2: Gamification characteristics and the LdoD Archive.

### 3.4 The LdoD Archive and the ESP game

In section 2.10, we introduced the ESP Game, its motivation, its main rules and ideas and its development throughout the years. Our goal now is to adapt this game’s ideas to our own problem.

The main goal of the ESP game was to address the issue of metadata. The game rules also involved not allowing user communication; enforcing specific times for the tasks and of course the game was
based on images that would be the object of classification.

After analysing the characteristics of the Archive and the Virtual Editions, our solution should be a game in which users classify text (fragments of a virtual edition); the tasks would be time limited as well; the players should not know who is currently participating in the game and our focus must be on promoting collaboration and the entertainment of our users and not only on the resulting output (the tag).

Our main takeways are that our archetype should be a crowdcreating process which, as stated in 2.7, is to create artifacts based on a diversity of heterogeneous contributions, and a game where users tag as groups is our solution. Beyond that, to motivate and show public recognition ranking and points are suggested to be implemented.

In the following section we present the game itself as our solution to the required collaboration.
The *LdoD* Classification Game

Contents

4.1 Game ................................................................. 32
4.2 Interface ......................................................... 32
4.1 Game

The LdoD Classification Game is a game that aims to use the Archive’s Virtual Editions and its ability to classify and categorize fragments as way to create crowdsourced tagged Virtual Editions in a fun and entertaining way.

A game is simply an online, real-time and synchronous environment, where a set of users meets and has the goal of categorizing a fragment of a selected virtual edition, resulting in an enriched virtual edition with new community categories, made in a collaborative and interactive way.

The game follows a mix of two different crowdsourcing processes that were introduced in 2.5. It consists in a crowdcreating process whose goal is to create artifacts based on a diversity of heterogeneous contributions and in a crowdvoting process where the wisdom of crowds is harnessed to filter the artifacts into one.

Nonetheless, how does this relate to the game itself? Well the game consists of three different stages, one in which each user individually suggests tags (crowdcreating), another where the player sees the suggestions of the other participants and can choose between maintaining his or her own suggestion or to changing the tag (voting other suggestion) and finally a more collaborative one, where all users see the votes and corresponding tags and as group vote in the best tag available for the fragment (a somewhat crowdvoting process).

The game can also be split into two different ways of creation: created by the Archive and custom. The first one is a game created by the LdoD Archive and abstracts all of the configurations necessary to create games. This mode is designed to be maintained and developed by the LdoD Archive organization by creating multiple games with the sole purpose of entertaining users. On the other hand, the custom game consists of a more private, personalized environment that can be co-related directly to the teacher-students scenario. Its allows for the design of a game for all registered users or only members of the same virtual edition (ideal for the classroom scenario); plan a specific time and date and play together.

4.2 Interface

As stated, the game takes inspiration from the ESP Game therefore the interface will have some similarities.

The game is initiated in a new console, which is opened from the main site, accessible only to the registered users of the LdoD Archive. As expected the users will have to log in on the game website in order to participate. The game platform has an About section where users will have the opportunity to re-read game rules, objectives and know more information about the game.

As mentioned above, the game can be split into different stages, hereby called rounds, that have different specific objectives in which the player must comply to. The fragment used in the game is split
into paragraphs and their size dictates the time available for each round and determines when the game should switch to a different stage.

The round follow a cycled approach, round one occurs for the first paragraph, round two occurs for the first paragraph. When it ends we switch to round one again however with the second paragraph of the fragment and so forth, until all of the paragraphs have been analysed in these two stages. Subsequently, we reach the final stage, round three, where the fragment text is used in full.

4.2.1 Round 1: Individually submit tags

In this round, the participants goal is to submit a tag for each paragraph of the fragment.

The user will have the first paragraph of the fragment to be placed at the center of the screen and bellow a category submission area. In the bottom part of the page, the user can see a progress bar showing visually how many paragraphs are still missing. At the top of the screen, in the center the user has a steps interface showing their current round and which round follows next. Above that and still in the center they can see the time remaining in the current round - note that the time is variable according to the size of the text. At the top left, the users can see the number of people participating online.

Summarizing, the player must in the given time, read the text carefully, think and submit one tag they believe is appropriate. The allowed input is equal to the input available in the LdoD Archive, which is anything above one character.

The time and occurrence in which a tag was suggested also plays a factor having impact in the calculation of scores. Let us consider player A suggests a category of dreams, also in the same stage a player B submits ten seconds later this same tag. Player A is considered the author of the tag and player B is a co-author or voter of the tag submitted of player A, this is important regarding the players scoring.

After the time elapses we move to the next step, which is the round 2 for the same paragraph. Note that the user returns to the round 1 after round 2 but with the next paragraph of the fragment.

In figure 4.1 we can see the interface for this round.

4.2.2 Round 2: Choose a tag

In round 2, the interface is very similar, showing again the same paragraph of the fragment, but now bellow, it will have the categories that have been suggested by all participants in the earlier round.

The goal in this stage is to once again find the best category available although now the user has the ability to choose from a range of options, as such the user can now choose/vote only in one category that the user agrees to be the most suitable for the paragraph that was analysed, the user can even choose the category that he/she submitted.

At this point the game becomes more interesting because the players now must simultaneously find a
category they seem fit while also trying to think which category will be chosen by the other participants. For round 2, independently of the size of the paragraph and since the players had earlier read and analyse the text, this round timer is of 30 seconds.

After the time expires, we move again to round 1 if there are remaining paragraphs to be analysed or otherwise we move to the final round.

The game interface for this round is shown in figure 4.2.

Figure 4.1: Game interface of round 1.

Figure 4.2: Game interface of round 2.
4.2.3 Round 3: Review and decide

After round one and two finish for all paragraphs, the challenge reaches the review deciding round.

The interface changes, the screen contains the complete fragment at the bottom part of the screen while in the top part there is a voting area containing the categories that won each voting. Additionally, we have a timer, and the indication that the game is in its final round.

This round is the final crowdvoting process, users can read if necessary the fragment as whole and the players must vote on one of the tags in order to elect the top tag of the fragment, which will be the final one and the one that gets to be included in the virtual edition. However this stage is more dynamic and iterative due to the fact that users now can switch their vote until the time expires whereas in the earlier stage they could only vote once on one tag. This round also has a fixed time, but of 45 seconds.

In figure 4.3 the final round is presented.

Accordingly to that they can also see the points of each tag and the current top tag in real time. Nonetheless the rule maintains that they can only vote in one category, which will be the option they leave selected when the time expires.

This presents the challenge of trying to simultaneously chose the best available tag whilst trying to predict the winner tag. The score of a participant follows the formula presented bellow:
\[
\text{Score} = s + s_{RWT} + v_{RWT} + s_{GWT} + v_{GWT} + c, \text{ where:} \\
\text{s: submit a tag} = 1 \text{ point;}
\]
\[s_{RWT}: \text{submit a round winning tag} = 5 \text{ points;}
\]
\[v_{RWT}: \text{vote on a round winning tag} = 2 \text{ points;}
\]
\[s_{GWT}: \text{submit the game winning tag} = 10 \text{ points;}
\]
\[v_{GWT}: \text{vote on the game winning tag} = 5 \text{ points;}
\]
\[c: \text{for each vote change} = \text{minus} 1 \text{ point;}
\]

From this we can see that, submitting a classification earlier is key (it allows to be considered the author of a tag); it is crucial to avoid making unnecessary vote changes and of course trying to find a tag that fits while trying to think in agreement with the other participants, resulting in the *wisdom of crowds*.

The player that suggested the winning tag will have their name credited as the author.

This formula tries to firstly encourage players to participate by giving each player that suggests a point. Furthermore, while someone may not vote or propose the winning classification, this formula rewards voting or suggesting classifications that win a round. Finally, an important point is the penalization of players that try to constantly switch their vote on the final round, by deducting a point for each switch. The reason behind this is to disincentive behaviour of not thinking and switching votes just because they believe it will be the winning classification. This way we believe the player will think before switching.
Contents

5.1 Server-side implementation ........................................ 38
5.2 Client-side implementation ......................................... 48
5.1 Server-side implementation

The server side was built upon the existing architecture of the LdoD Archive. The Archive is based on Java, the Fénix Framework\(^1\), the Spring Framework\(^2\) and JavaServer Pages (JSP).

As expected, Java is used for the business logic while the JSP is used for the presentation logic of the Archive.

In order to have a persistent domain model the Archive uses the Fénix Framework, which is an open-source Object-Relational Mapping (ORM) developed at Instituto Superior Técnico (IST). The framework uses its own language Domain Modelling Language (DML) and overall the use of this framework simplifies the support of transactional behavior and persistence.

On the other hand, the Archive uses the Spring framework, which provides many other modules for Java based application that also make everything easier, such as Spring Model View Controller (MVC).

5.1.1 Domain Model

The new domain model is simply an extension of the existing LdoD Archive domain model. This model is presented as a Unified Modeling Language (UML) Class Diagram in figure 5.1, notice that this diagram is simplified as a way to make it easier to understand and follow, showing only the classes that matter for the game interaction and without their attributes.

As we can observe from the figure, the new domain model has four new entities: ClassificationGame, ClassificationGameParticipant, ClassificationGameRound and Player. Player is a very simple entity that only holds score as an attribute and has two relations one with a LdoDUser and another with the ClassificationGameParticipant. The Player semantic is evident: an LdoDUser can be a Player or not (0..1) and a Player can only be associated with an LdoDUser; also a Player can be a Participant in many games or none at all and the inverse is also true. This allows to create a game without initially defining who plays and giving the flexibility to allow for adding more players later, but only if the game hasn’t started yet.

On the other hand, ClassificationGame is a bit more complex having mainly a description, a datetime corresponding to the moment in which a game should be deployed, a sync variable that signalizes if the game must be synced (developed in section 5.1.7) and finally has a state machine with the following states CREATED, OPEN, STARTED, TAGGING, VOTING, REVIEWING, ABORTED, FINISHED. These states as the name suggest are applied to each point the game is in, the more pertinent to note are OPEN, STARTED, ABORTED and FINISHED. When the date to which the game was scheduled arrives, the game moves to a OPEN state that means that the game is now waiting for users to connect. Subsequently, if the game has at least two participants the game moves to STARTED otherwise is

\(^1\)https://fenix-framework.github.io/
\(^2\)https://spring.io/
ABORTED. Afterwards, when the game is completed it changes to FINISHED. Finally, it is important to remember that any ClassificationGame complies with the rules of the Virtual Edition that is associated with, as stated in 3.1.

Lastly, the ClassificationGameRound allows to keep a history of interactions between the participant and the game, having information like in which rounds the user played (attribute round), what did he submit; vote; at what time and in which paragraph (identified by the variable number).

The other relations are explained in more detail bellow, having the following semantics:

- **Game - Virtual Edition**: A game must be associated with a virtual edition and a virtual edition can have many games, this relation gives access to relevant information like which users are members of the virtual edition;

- **Game - VirtualEditionInter**: The game must have one fragment (which of course belongs to a virtual edition) to be used in the game and said fragment can be used in various games;

- **Game - LdoDUser**: A LdodUser must be a game owner/responsible (which is the person that generated the game) and said user can also own other games, this relation allows the management of the game if need be. The game owner does not need to be one of the game’s players;

- **Game - Tag**: A game produces a Tag, this translates to the game objective of tagging a fragment. The relation is 0...1 regarding the tag because at the moment of the game creation we do not know which Tag will be produced and, on the other direction, a tag may not be related to a game.

- **Game - Participant**: A game can have multiple participants, however a participant is associated with only one game. Notice that participant is different entity from a player. A player plays in different games and it can be seen like a universal entity that relates with multiple games while a participants is a local view within each game.

The corresponding DML code for the extended domain can be found in appendix A.

### 5.1.2 REST API

This section describes the Representational State Transfer (REST)ful Application Program Interface (API) that was developed to support the communication between the server (the LdoD Archive) and our client (the Classification Game). The application developed uses aforementioned Spring Framework, which allows the definition of one or more Java classes to act as Controllers and handle requests.

Defining a controller in Spring is easy as annotating it with the @Controller annotation, or in our case @RestController. Four main controllers are of interest to us: the Authentication Controller; the API User Controller; the API VirtualEdition Controller and the ClassificationGameController.
Figure 5.1: Enriched UML Class Diagram of the LdoD Archive with Classification Game entities
5.1.3 Authentication Controller

This controller is only responsible to authenticate users from outside applications that connect to the server. It is the only endpoint that is truly open to everyone since it is not covered by a JavaScript Object Notation Web Token (JWT) filter that checks the token validity.

<table>
<thead>
<tr>
<th>Name</th>
<th>Request Method</th>
<th>Endpoint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuthenticateUser</td>
<td>POST</td>
<td>/api/auth/signin</td>
<td>Using the Archive username and password credentials, authenticate the user and return to the client a JWT that allows connection to other endpoints.</td>
</tr>
</tbody>
</table>

Table 5.1: API Authentication Controller documentation.

5.1.4 User Controller

As the name suggests operations related to a user are mapped to this controller. This controller returns a Data Transfer Object (DTO) of an user in its methods.

<table>
<thead>
<tr>
<th>Name</th>
<th>Request Method</th>
<th>Endpoint</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CurrentUser</td>
<td>GET</td>
<td>/api/user</td>
<td>Returns the current logged in user.</td>
</tr>
<tr>
<td>UserProfile</td>
<td>GET</td>
<td>/api/user/{username}/</td>
<td>For the given username parameter the server returns their corresponding DTO .</td>
</tr>
</tbody>
</table>

Table 5.2: API User Controller documentation.

5.1.5 Virtual Edition Controller

All of the endpoints for this controller start under the `api/services/` path. The requests of this controller are subject to control permissions. A user can only inquire about his/her own virtual editions and must be logged in. This is achieved using the following annotation `@PreAuthorize("hasPermission(#username, 'user.logged')").`

On the other hand, public virtual editions can be queried by anyone and this is done by using the annotation: `@PreAuthorize("hasPermission(#acronym, 'editionacronym.public')").`

Table 5.3 contains the REST methods available for this controller.

5.1.6 Classification Game Controller

This controller it is perhaps the most important from all of the above, it manages all the game interactions in two different ways: REST requests and real-time communications using Streaming Text Oriented Messaging Protocol (STOMP) over WebSocket\(^3\). while also saving data to the database.

Regarding the Restful requests we have:

\(^3\)https://stomp.github.io/stomp-specification-1.2.html.
Concerning STOMP, it has identical characteristics to Hypertext Transfer Protocol (HTTP). Uses Transmission Control Protocol (TCP) along with providing a set of commands, yet we will focus on the most relevant to the scope of this thesis which are: CONNECT/DISCONNECT; SUBSCRIBE/UNSUBSCRIBE and SEND.

The choice to use STOMP is by virtue of Spring supporting this protocol which makes it very easy to setup, as we can see below:

```java
@Configuration
@EnableWebSocketMessageBroker
public class WebSocketConfig extends AbstractWebSocketMessageBrokerConfigurer {

    @Override
    public void registerStompEndpoints(StompEndpointRegistry registry) {
        registry.addEndpoint("/ws").setAllowedOrigins("*").withSockJS();
    }

    @Override
    public void configureMessageBroker(MessageBrokerRegistry brokerRegistry) {

    }

    public void registerStompEndpoints(StompEndpointRegistry registry) {
        registry.addEndpoint("/ws").setAllowedOrigins("*").withSockJS();
    }

    public void configureMessageBroker(MessageBrokerRegistry brokerRegistry) {

```
brokerRegistry.setApplicationDestinationPrefixes("/ldod-game");
brokerRegistry.enableSimpleBroker("/topic/ldod-game/");
}
}

Listing 5.1: WebSocketConfig.java

What this code does is to setup an endpoint at the specified Uniform Resource Locator (URL) and allows any origin over SockJs. It also configures the broker while specifying that client subscriptions should be made under the /topic/ldod-game/{gameId} URL and the client should send messages with the starting prefix of /ldod-game/{gameId}.

After this, the usage is very straightforward:

@Autowired
private SimpMessagingTemplate broker;

(...)

private void startGame(String id) {
    Map<String, String> payload = new LinkedHashMap<>();
    payload.put("currentUsers",String.valueOf(playersInGame.size()));
    payload.put("command", "ready");
    broker.convertAndSend("/topic/ldod-game/" + id + "/register", payload.values());
}

Listing 5.2: Usage example of a STOMP broker.

In this code, a broker is instantiated and we have a simple method that sends to the subscribed users of the topic register that a game can start. The corresponding client interaction can be found in listing 5.7.

Regarding the server STOMP related handler methods, in table 5.5 we have a summary, notice that all communications use a gameId in the URLs which ensures that clients and server communicate only with the specific users of a game and not the whole universe of other players. All these methods mentioned, do a transaction to save into the database data such as tags; votes; rounds and scores.

5.1.7 Synchronization

The synchronization aspect of the game is mainly done by the entity GameRunner (5.1.9) with complementary help of a class that schedules tasks (5.1.8) and the ClassificationGameController (5.1.6).
<table>
<thead>
<tr>
<th>Name</th>
<th>Receive</th>
<th>Send Endpoints*</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect</td>
<td>/connect</td>
<td>/config</td>
<td>Simple method that registers users as participants in the database.</td>
</tr>
<tr>
<td>Tags</td>
<td>/tags</td>
<td>/tags</td>
<td>Method that stores locally relevant data (player score, author, tag points) and sends it back.</td>
</tr>
<tr>
<td>Votes</td>
<td>/votes</td>
<td>/votes</td>
<td>Identical to the previous method but with appropriate semantics for a vote.</td>
</tr>
<tr>
<td>Review</td>
<td>/review</td>
<td>/review</td>
<td>This method is called before advancing to the final round, finds which tags are at the top and sends them to the clients.</td>
</tr>
<tr>
<td>Sync</td>
<td>/sync</td>
<td>-</td>
<td>This method is called in order to request the server to perform a sync stage of the game. It simply changes the variable sync of the ClassificationGame to true. GameRunner is responsible to respond when the synchronization has ended.</td>
</tr>
</tbody>
</table>

* The receiving endpoints follow the base URL of `/ldod-game/{gameId}` and the sending endpoints the `/topic/ldod-game/{gameId}`.

Table 5.5: Classification Game Controller STOMP methods.

Figure 5.2: State machine that illustrates the synchronization flow of a game.

As previously mentioned, the ClassificationGameController the method sync changes the game into a sync stage. This is dealt by the GameRunner that periodically checks for this information and when it finds that the game is supposed to sync it will wait for the clients to reach this stage and then tell them to resume the game.

The main concept for the synchronization of the players and the game is to make the game clients alert the server each time they have completed something. From there, the server wait some time for the clients and afterwards informs them to proceed the game. In the time, that the server is waiting the clients stop and do not nothing while holding for a command to continue.

In figure 5.2, this is explained, we can see the flow and states that a game suffers throughout time.
5.1.8 Scheduled Tasks

This class is responsible to wake up instances of Game Runner. It executes every minute a method that finds which games are available (getGames()) and creates a new thread of Game Runner for each available game. The getGames method simply reads from the database which games are in a CREATED state and that are scheduled to start in the next two minutes:

```java
@Component
public class ScheduledTasks {

@Autowired
GameRunner gameRunner;

@Scheduled(cron="0 * * * * *")
public void scheduleGames() throws IOException {
    List<String> gameIds = getGames();
    for (String id: gameIds) {
        gameRunner.setGameId(id);
        new Thread(gameRunner).start();
    }
}
(...)
```

Listing 5.3: ScheduledTasks excerpt related with Game Runner.

5.1.9 Game Runner

Firstly, in order for this class to be able to send messages through the web socket to clients, some configurations had to be made. As such the class was annotated with @Component and a web socket broker was injected as a constructor dependency. Furthermore, since we want the ability to have multiple games occurring at the same time, our system must create a thread for each game and of course each game must have its own context. Spring provides this scope 4, however it does not include it by default, so we registered this scope in the application. After this configuration, we simply annotate the Game Runner class with @Scope("thread").

```java
public class CustomScopeRegisteringBeanFactoryPostProcessor implements BeanFactoryPostProcessor {
    @Override
```

4https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/context/support/SimpleThreadScope.html/
public void postProcessBeanFactory(ConfigurableListableBeanFactory beanFactory) throws BeansException {
    beanFactory.registerScope("thread", new SimpleThreadScope());
}
}

Listing 5.4: Creating a scope called “thread” of the SimpleThreadScope class from Spring.

@Configuration
public class CustomScopeConfig {

    @Bean
    public static BeanFactoryPostProcessor beanFactoryPostProcessor() {
        return new CustomScopeRegisteringBeanFactoryPostProcessor();
    }
}

Listing 5.5: Registering the “thread” scope created above.

Apart of all this initial configuration, the Game Runner which implements the Runnable interface, relies on its main method, the run method (overridden from the Runnable interface). It acts as a decider for each given conditions, calling all of the bellow methods. Firstly, if the games has entered the OPEN state, then keeps checks the current time and the game starting time and when it has reached one minute after the starting time it starts (this one minute windows allows users to connect).

After this, the GameRunner checks if the game can start (i.e, the number of users that connected is at least 2). If the game proceeds, the GameRunner is in a loop until the game has not finished, and periodically checks if the game must be synced (this is triggered by the ClassificationGameController that it puts the sync variable to true when the clients request it). Then the GameRunner, waits some time for the users to reach this stage and after that it sends to the clients a command to continue the game.

Bellow a simple specification of each method called by run:

- **canGameOpen**: Checks if the id given corresponds to a created game, if so transitions the game to the OPEN state otherwise returns false;

- **canGameStart**: See how many users have connected, if there at least two the game can start otherwise return false;

- **canGameContinue**: Tests to see if the sync variable is true or not;

- **hasGameEnded**: Queries the game state to see if it is in a FINISHED state;
• **startGame**: Switches the game to the **STARTED** state and informs clients that the game will start;

• **abortGame**: The game is swapped to **ABORTED** and clients receive this information;

• **continueGame**: Changes the game sync variable to false and issues a command to all clients to continue the game.

```java
@Component
@Scope("thread")
public class GameRunner implements Runnable{

    private static Logger logger = LoggerFactory.getLogger(GameRunner.class);
    private String gameId;
    private DateTime startTime;
    private Set<String> playersInGame;

    @Autowired
    private SimpMessagingTemplate broker;

    @Autowired
    public GameRunner(final SimpMessagingTemplate broker) {
        this.broker = broker;
    }

    public void setGameId(String gameId) {
        this.gameId = gameId;
    }

    @Override
    public void run() {
        if (canOpenGame()) {
            while (Instant.now().isBefore(startTime.plusMinutes(1))) {
                try {
                    Thread.sleep(600);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
            if (canGameStart()) {
                startGame();
                while (!hasGameEnded()) {
                    if (canGameContinue()) {
                        try {
                            Thread.sleep(600);
                        } catch (InterruptedException e) {
                            e.printStackTrace();
                        }
                    }
                }
            }
        }
    }
}
```
Listing 5.6: GameRunner code excerpt.

5.1.10  *LdoD Archive*: Game Interface introduction

On the server side some UI changes were made in order to support the game and game elements. The *new LdoD Archive* interface now includes: a game creation/management web page; a game leaderboard web page; a user profile with new information such as points and overall position in the ranking. Figure 5.3 shows the changes added.

The game environment is accessible from the Virtual Editions page. All of this was made using JSP for the view logic, while being supported with controllers code to provide the data.

5.2  Client-side implementation

The client for the game was built using ReactJS which is an open-source JavaScript (JS) framework maintained by Facebook and it is usually used to develop UIs.

The framework main concept revolves around Components, which can be compared as Java classes. A component is something that outputs HyperText Markup Language (HTML) while having every piece of data necessary to present that HTML, components can also maintain state. Furthermore, components have JS functions necessary to make it dynamic. Components also have methods that determine their life cycle and allow the developer to control better the presentation. Another important concept is Props which are the way components communicate with each other, passing data between them.

In the following subsections we present our main components and their importance to the game and server communication.
(a) Game environment access.

(b) Game management area.

Figure 5.3: New game interface features in the *LdoD Archive*. 
5.2.1 Game Components

The game has four main components that focus on getting data from the server and deciding what to do next: App, Game, Virtual Edition and Fragment. While we also have four other components mostly focused on game mechanics and presentation: Paragraph, Tag, Vote, Review. A diagram representing these components and how they relate can be found in figure 5.4.

Our main component is the App component responsible for authenticating a user; fetching active games for the logged in user and registering the paths for the other components.

Afterwards, we have the Game component that is responsible for connecting and registering users to a game; deciding if the game starts or aborts accordingly to the command sent by the server and rendering the component Virtual Edition. A Virtual Edition component fetches the fragment to use and passes it to the Fragment component, which in turn splits the fragment text into paragraphs and passes it along to the Paragraph component.

A Paragraph can then render different components according to each stage of the game: Tag (Round 1), Vote (Round 2) and Review which renders a Vote because reviewing also includes voting (Round 3).

Each of these last three mentioned components communicate with the server through the use of Websockets, as explained in 5.2.2.

These components are very similar in their behaviour and communication. Firstly, as aforementioned the Paragraph is the component which determines everything about these components: the Paragraph holds the text for a paragraph of the Fragment selected to use in the game and the life cycle of a Paragraph is determined by the time attributed.
Longer paragraphs correspond to more time to read; submit a tag and vote. So for each Paragraph for the specified time: renders a Tag (which simply allows for text input; saves data submitted and transmits to the server); following this a Paragraph requests a server sync; moves to render a Vote component (similarly allows for voting input; save vote date and relays to the server), this is repeated for all paragraphs until when we reach the final paragraph after which we render a Review component that receives the fragment text, requests the server which tags are to be voted and finally renders a Vote component which acts similarly as before but now allows vote changes. Subsequently, when the time expires the Paragraph component uses a callback to alert the Virtual Edition component that the games has ended; which in turn alerts the server and the server replies with the game winner.

To note, the Virtual Edition component could be removed and the Game would communicate directly with the Fragment because games at the moment are only for one fragment. However, this allows for a easier way to extend the game. In another iteration of the game if we would want to have multiple fragments to be used it would be a simple as making the Virtual Edition fetch them all and control the render of each Fragment in turns.

5.2.2 Websockets

In order to communicate with the server in real-time, as aforementioned the solution was to use Websockets over STOMP. The decision to use also is due to the easiness to setup and start to use. For this, react-stomp was used, which is React component for SockJS-client. The usage is effortless as:

```javascript
import SockJsClient from 'react-stomp';
import WEB_SOCKETS_URL from 'Constants';
class Game extends Component {
  constructor(props) {
    super(props);
    this.connect = this.connect.bind(this);
    this.onMessageReceive = this.onMessageReceive.bind(this);
  }

  componentDidMount(){
    (...)
    this.setState({
      (...)
      socket:
```
```jsx
<SockJsClient
  url={WEB_SOCKETS_URL}
  topics={[ '/topic/ldod-game/' + gameId + '/register' ]}
  ref={ (client) => { this.clientRef = client }}
  onConnect={ () => { this.connect(gameId) }}
  onMessage={ (message) => this.onMessageReceive(message) } />

connect(gameId){
  (...)
  this.clientRef.sendMessage('/ldod-game/' + gameId + '/register', JSON.stringify({
    userId: this.props.context.currentUser.username, gameId: gameId}))
}

onMessageReceive(message) {
  (...)
  if(command === "ready"){
    this.setState({
      currentUsers: users,
      isLoading: false,
    })
  }
  else if(command === "aborted") {
    this.setState({
      isAborted: true,
    })
  }
}
```

Listing 5.7: Usage example of react-stomp.

In this code excerpt, we have a component through a web socket subscribes to register topic and when the web socket connects to the server it sends a userId and gameId. As a callback the client indicates that messages received from the server should be handled by the onMessageReceive method. This method checks if the server requested for the game to proceed or abort.
5.2.3 External Libraries

Besides all of the mentioned above, the React client developed uses other external libraries of relevance, one of them was already mentioned (react-stomp). The others are react-bootstrap\(^7\), antd\(^8\) and react-countdown-clock\(^9\).

The usage of this libraries is mostly related with the UI. React-bootstrap for instance is simply a rebuild of the popular Bootstrap framework for React and in order to maintain some consistency between the look and feel of both the server and the client it made sense to use Bootstrap as well. The antd library is useful for some icons and other interface objects not available in Bootstrap. While on the other hand, react-countdown-clock is mainly for being a simple and yet an appealing and customizable countdown timer.

\(^7\)https://react-bootstrap.github.io/
\(^8\)https://ant.design/
\(^9\)https://github.com/pughpugh/react-countdown-clock
Results

Contents

6.1 Interface Auto-Evaluation through Heuristics ............................ 55
6.2 Game Evaluation through User Feedback ................................. 57
6.3 Tag Evaluation through Expert Review ................................. 60
6.1 Interface Auto-Evaluation through Heuristics

In this section, we use Nielsen's usability heuristics [27] as a guideline to understand if our UI complies with these heuristics and in which way, and if not what should be made in order to fix them.

Below we have a list of the ten heuristics proposed by Jakob Nielsen:

1. **Visibility of system status:** *The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.*

   This heuristic is supported in our system through the following features:

   - Progress Bar: each game has a progress bar that visually indicates how many paragraphs are still remaining;
   - Steps: every game presents a step by step feature, illustrating in which current step the game is in (one - tagging; two - voting; three - reviewing);
   - Menu Header: a common feature in every interface, a menu allows the user to show where the user can go to;
   - Timers: for each game step, there is also a timer specifying the available time for the task at hand.

2. **Match between system and the real world:** *The system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.*

   Our platform adheres to this heuristic by mainly using the same terms presented in the Archive, even though some new concepts are introduced (due to being game specific) they all are natural terms (*vote, submit, tags* are some of the game terms which are pretty self-explanatory).

3. **User control and freedom:** *Users often choose system functions by mistake and will need a clearly marked “emergency exit” to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.*

   This heuristic is not supported by our platform at the moment the game is built around definitive decisions (the exception being the final stage of the game where users can change their votes). Unfortunately, at this point if a user votes or submits something by mistake the game takes it as final. As such this heuristic reveals a need to fix this situation in the future.

4. **Consistency and standards:** *Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.*
This heuristic is respected as the game tries to follow the same conventions and words that are used in the LdoD Archive. While also trying to preserve a presentation that has some similarities with the Archive in order to be easier to follow.

5. **Error prevention**: Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

This heuristic is very much related with heuristic number three, and as stated before, this principle is not respected since our users are not able to prevent error.

6. **Recognition rather than recall**: Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Through the use of progress bar; menu header; steps; timers and other features like constantly showing what tags has the user submitted we believe this principle is respected. However this is something that needs to be verified by constant system evaluation.

7. **Flexibility and efficiency of use**: Accelerators—unseen by the novice user—may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

This principle is not fulfilled nevertheless this is mainly due to game restrictions, considering we want users to always be in sync while the game occurs so no accelerators are provided.

8. **Aesthetic and minimalist design**: Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

We believe our interface also complies with this heuristic, firstly our design tries to maintain some similarities with the ones existing in the Archive in order to make it easier to follow, this for instance can be seen by the use of the Bootstrap library in both platforms. Despite some differences from the LdoD Archive, it also shows the same kind of minimalist design.

9. **Help users recognize, diagnose, and recover from errors**: Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

We can say this heuristic is partially obeyed, since some error messages are presented (login problems; server related problems and game related problems), however their language and path
to a solution are not always presented. While a login problem instructs a user to check his/her credentials a down server or fetching problems are not explained with much detail, simply a message appears.

10. **Help and documentation:** *Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user’s task, list concrete steps to be carried out, and not be too large.*

The game platform contains an About page with a Frequently Asked Questions section, which tries to inform the user on game related question. Furthermore, we think that this web page must be constantly updated so the users can ask more questions (that could later be added if deemed relevant) by using the Feedback page of the game website.

Appendix C contains game screens that show interface characteristics mentioned above (progress, steps, timers, game messages, error page, among others).

### 6.2 Game Evaluation through User Feedback

The accomplishment of any product frequently relies on assessment of the general population that will use it. In this case, if a game is alluring, entertaining and addictive it will probably be exceedingly utilized.

The developed game endeavors to make people collaborate with each other in a fun manner in order to produce new artifacts (Virtual Editions with new tags) whilst promoting knowledge about a literary work by Fernando Pessoa.

As the fundamental objective of the created application is to be a text based game, it was asked to colleagues; family members; existing users of the *LdoD Archive* to test the prototype, in order to better understand its positives and also negatives.

After testing the game prototype, participants were requested to fill a little questionnaire that assess and profiles the user. Part one of the questionnaire focuses on group age; gender; acquaintance with the work of Fernando Pessoa and gaming habits. While the second part targeted characterization of the game itself such as its ease of understanding, rules and entertainment.

The full questionnaire and complete charts can be seen in Appendix D.

The questionnaire had sixteen responses. Thus, every interpretation we take from this must be weighted and cannot be extrapolated without having more responses.

Regarding the profiling of the inquired we can say that nine of the survey respondents are men and seven women; the majority of the participants have an age between 19 and 24 and all of the inquired have a higher education degree (the majority, 10 a graduation). When questioned about knowing about
the Book of Disquiet only two said that they did not had heard of its existence. Furthermore half of the people had read the book and also half selected I am a normal reader of Fernando Person knowing only mandatory works as the option that best described them. Five considered themselves as an above average reader and three as experts. Finally, only one person never plays games and a majority (five people) play games daily.
Analysing this section of the questionnaire, firstly its obvious that it is a very small sample size. The majority of users are very young; know some thing about Fernando Pessoa and regularly play games.

The next section, focused on finding about the game experience the players had. In a scale from 1 to 5 (very easy, easy, normal, hard, very hard) users had to rate the game regarding the difficulty of understanding it (objectives and rules). The results were: seven choose rating very easy, two choose rating easy, three normal, three hard and one found it very hard. Regarding the entertainment that the game provided to the participants, from 1 - not fun to 5 - very fun, two people found it not fun while a majority (8) found it fun and six other found it normal.

The majority (eleven people) would consider playing again and would recommended the game to another person. Two people said they would not play again and five responded as maybe regarding recommending the game. The question if the game made it interesting to read and learn more about the Book of Disquiet and Fernando Pessoa, was very divisive with seven people saying yes, a lot, other four not at all and the rest in between with a bit.

So from this, considering this small sample size we can say that, for now, the game seems to be easy to follow; it has not reached a high level of entertainment; it has the potential to entice players to play again and could also attract other users to join. On the other hand, it does not seem to push and promote the interest on Fernando Pessoa and the book.

After this, the questions were of open format and not mandatory. The questions were what did the user like the most and the least; what did the user think it should be improved and any other suggestions.

The most relevant points brought up for each of these questions were:

- most liked: compare what other participants wrote; originality; reading and knowing fragments of
the book in a fun manner;
• least liked: time pressure;
• improvements: interface related changes (text size, formatting and UI in general); weekly rankings, increase time, in the end showing what tags other users in other games had chosen for that fragment;
• general suggestions: ranks and social achievements; increase times; provide suggestions of classifications in round 1; explain better the game (to the general audience and how a game must be created and its rules).

In conclusion, noting the small sample size this questionnaire confirms our need to keep having feedback of our players in order to improve. The game as is has some potential, but it must follow the user base suggestions in order to reach its goals which at the moment it does not.

6.3 Tag Evaluation through Expert Review

Finally, another way to understand our results was to consult with an expert on Fernando Pessoa’s work by requesting to determine if the produced tags were good and adequate for the fragments in question.

For this, Dr. Manuel Portela was invited to analyse if the yielded results could be considered suitable. The main conclusions presented by professor Manuel Portela were as follows:

<table>
<thead>
<tr>
<th>Number</th>
<th>Title</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>472</td>
<td>Omar Khayyam</td>
<td>joy</td>
</tr>
<tr>
<td>82</td>
<td>Irrita-me a felicidade de todos estes homens</td>
<td>vain philosophy</td>
</tr>
<tr>
<td>60</td>
<td>Senti-me inquieto já</td>
<td>restlessness</td>
</tr>
<tr>
<td>51</td>
<td>Trez dias seguidos de calor sem calma</td>
<td>After the storm comes the calm</td>
</tr>
<tr>
<td>43</td>
<td>O que ha de mais reles nos sonhos</td>
<td>Indecision</td>
</tr>
<tr>
<td>31</td>
<td>“Sentir é uma macada”</td>
<td>Boredom of life</td>
</tr>
<tr>
<td>14</td>
<td>Afinal deste dia fica</td>
<td>Insatiable Anguish</td>
</tr>
</tbody>
</table>

Table 6.1: Fragments and its tags.

The analysis of this set of labels shows two strategies of semantic analysis of the text: on one hand, the identification of keywords (or key phrases) in the text itself, making the topic a citation of a word or passage from the text; on the other, the abstraction of a general topic understood as a dominant topic but which is expressed in words different from those that appear in the text. The labels tend towards an abstract topicalization of the text that loses the reference to the concrete context of the narrator’s reflection - this characteristic is one of the difficulties that the fragments of the Book of Disquiet place on any definition of its topics, as texts always oscillate between a metaphysical / abstract reflection of the narrator and a concrete observation of his daily life. The labels reveal this tendency to an overly generic abstraction (for example, “joy” for the text “Omar Khayyam” seems to completely omit the topic of drinking
wine as a celebration of life’s ephemerality or the topic of life’s boredom.) Probably with more players
categories would have emerged. On the other hand, these labels do not
stray far from those that many people would create.

Professor Manuel Portela also suggested to consider a new game version that would request players
to define two tags: one more abstract (focused on thoughts and feelings) and one more specific (focused
on the narrator’s observations). Furthermore, the voting would also be for these two groups, meaning
that an user would vote on a tag for the abstract group and another for the more specific one. This would
avoid too generic tags, producing more powerful and accurate results.

After analysing this review, we can understand that although the produced artifacts are acceptable
and do not stray from the norm of expected tags, it could be improved by introducing this new version of
two group tags.
Conclusions

Contents

7.1 Summary of the Dissertation ........................................... 63
7.2 Contributions ................................................................. 63
7.3 Future Work ................................................................. 64

62
7.1 Summary of the Dissertation

The motivation behind this dissertation was to considering the existing context of the Book of Disquiet and the LdoD Archive functionalities to develop a solution that promotes interaction of the Archive’s users. Therefore, we defined as our goal to use the Archive’s Virtual Editions as a tool that allied with crowdsourcing and gamification could increase the collaboration of the users.

Our work consisted firstly of researching existing solutions of crowdsourcing and gamification within the context of Digital Humanities, that could serve as a reference point to our solution. From this research, we formulated the idea to develop a solution that would be inspired by the ESP game, that would use a crowdvoting mechanism and would have common game elements like points and a leaderboard. We also analysed the Ldod Archive and its Virtual Editions, resulting in using the existing classification mechanism in the context of our game.

From a more technical point of view, the developed work occured in two ways: server-side and client-side code. On the server-side, our domain was extended, new controllers for REST communications were added as well as a configuration for websockets. Additionally, new classes were developed to allow the game synchronization. Our client-side code, was built using React and provides the game interface. While also, providing the necessary server communication.

Finally our work was analysed from three perspectives: a heuristics analysis, in order to understand the validity of the interface; a user evaluation through a questionnaire and an expert review of the produced artifacts.

In the following sections we present our contributions and achievements and what should be made in future developments.

7.2 Contributions

The accomplishment of any social oriented tool is of course very dependent on its user base and their interactions with one another. As such the platform’s success will be a constantly evolving accordingly to the user’s suggestions and improvements.

However, there are already some visible results. Firstly, a foundation for an API, that includes adequate JWT login; RESTful methods and Websockets communications, was developed and can be now extended to anything that it may seem relevant and interesting in the future. Additionally, this thesis allowed to further expand the existing features of the LdoD Archive and integrate them in a new environment.

Beyond this, our prototype allowed and paved the way to integrate more outside environments with the current platform, since the new LdoD Archive now serves two interfaces one from JSP and other from the React game.
Regarding the game, despite the fact that the number of users to test the game at the time of the writing of this thesis is undoubtedly low, we can already take some interesting points. As stated in 6.2, the game already has some solid synchronization without having any reported incidents. The majority of the people found the game very easy to follow and a majority voted as entertaining. On the other hand, the responses reflect a mandatory need to use this feedback in the future to improve the game and reach its potential.

7.3 Future Work

The future work on this solution should be based mostly on user feedback focusing on many different aspects in order to achieve the best possible engagement.

This should address the following aspects: point distribution; changes to the current game flow; fine tune the available time for each paragraph in order to reach an optimal time; introduce thresholds and taboo words for saturated words; UI modifications for having a modern game like experience; although the game is responsive for mobile it could also be expanded to a mobile app which is a platform very appealing for quick games; new game modes and mainly include community suggestions that seem to be in the interest of a large majority of the playing user base.

An idea for the future would be to develop a game with a slightly different goal, that would focus on a synthesis based on paragraphs rather than the fragment. This new game version instead of having only one tag by fragment, would have one for each paragraph. This could also reinforce the collaborative aspect of the game by allowing multiple winners. Additionally, as mentioned in the expert review made by Professor Manuel Portela a new game version could include two tags by fragment that are grouped by an abstact tag and a specific tag.

On the other hand, the Archive website could have a reformed UI that revamp existing elements and introduce other such as profiles; leaderboards; social badges and groups; and other elements to make a bridge between the game and the website.

Furthermore, the game should be internationalized, as it is a very important factor, making it a primary step to be considered because it enables to reach a larger crowd.

In order to keep track of the project evolution, the source code is publicly available in a GitHub repository¹ and the game can be accessed through the link present in the webpage² or directly at the game homepage³.

---

¹https://github.com/socialsoftware/edition
²https://ldod.uc.pt/virtualeditions
³https://ldod.uc.pt/classification-game
Bibliography


Domain Modelling Language
Listing A.1: DML for the extended domain.

class ClassificationGame {
    ClassificationGameState state;
    String description;
    boolean openAnnotation;
    DateTime dateTime;
}

class Player {
    double score;
}

class ClassificationGameRound {
    int number;
    String tag;
    DateTime time;
}

class ClassificationGameParticipant {
    boolean winner;
    double score;
}

relation VirtualEditionHasClassificationGames {
    VirtualEdition playsRole virtualEdition {multiplicity 1..1;}
    ClassificationGame playsRole classificationGame {multiplicity 0..*;}
}

relation LdoDUserOwnsClassificationGame {
    ClassificationGame playsRole responsibleForGames {multiplicity 0..*;}
    LdoDUser playsRole responsible {multiplicity 1..1;}
}

relation ClassificationGameForVirtualEditionInter {
    ClassificationGame playsRole classificationGame {multiplicity 0..*;}
    VirtualEditionInter playsRole virtualEditionInter {multiplicity 1..1;}
}

relation LdoDUserIsPlayer {
    LdoDUser playsRole user {multiplicity 1..1;}
    Player playsRole player {multiplicity 0..1;}
}
relation ClassificationGameProducesTag {
    ClassificationGame playsRole classificationGame \{multiplicity 0..1;\}
    Tag playsRole tag \{multiplicity 0..1;\}
}

relation ClassificationGameParticipantIsPlayer {
    ClassificationGameParticipant playsRole classificationGameParticipant \{multiplicity 0..*;\}
    Player playsRole player \{multiplicity 1..1;\}
}

relation ClassificationGameParticipantInGame {
    ClassificationGameParticipant playsRole classificationGameParticipant \{multiplicity 0..*;\}
    ClassificationGame playsRole classificationGame \{multiplicity 1..1;\}
}

relation ClassificationGameParticipantInGameRounds {
    ClassificationGameParticipant playsRole classificationGameParticipant \{multiplicity 1..1;\}
    ClassificationGameRound playsRole classificationGameRound \{multiplicity 0..*;\}
}
Project code
Listing B.1: ClassificationGameController.java

```java
@RestController
@RequestMapping("/api/services/ldod-game")
public class ClassificationGameController {

    private static Logger logger = LoggerFactory.getLogger(ClassificationGameController.class);

    @Autowired
    private SimpMessagingTemplate broker;

    private final Map<String, ClassificationGameDto> gamesMapDto = new LinkedHashMap<>(100);

    // ------------- REST Methods ------------- //

    @GetMapping("/{username}/active")
    @PreAuthorize("hasPermission(#username, 'user.logged')")
    @ResponseBody
    public ResponseEntity<List<ClassificationGameDto>> getActiveGames(@PathVariable(value = "username") String username) {
        // logger.debug("getActiveGames: ", username);

        List<ClassificationGameDto> result = LdoD.getInstance().getActiveGames4User(username).stream()
            .map(ClassificationGameDto::new)
            .sorted(Comparator.comparingLong(ClassificationGameDto::getDateTime))
            .collect(Collectors.toList());

        for (ClassificationGameDto gameDto : result) {
            if (!this.gamesMapDto.containsKey(gameDto.getGameExternalId())) {
                this.gamesMapDto.put(gameDto.getGameExternalId(), gameDto);
            }
        }

        return new ResponseEntity<>(result, HttpStatus.OK);
    }

    @GetMapping("/end/{gameId}")
    public @ResponseBody ResponseEntity<?> end(@PathVariable(value = "gameId") String gameId) {
        // logger.debug("end: ", gameId);

        ClassificationGame game = FenixFramework.getDomainObject(gameId);
        game.finish();

        String usernameWinner = game.getClassificationGameParticipantSet().stream()
            .filter(ClassificationGameParticipant::getWinner)
            .findFirst().get().getPlayer().getUser().getUsername();

        List<Object> response = new ArrayList<>();

        return new ResponseEntity<>(response, HttpStatus.OK);
    }
}
```
response.add(usernameWinner);

return new ResponseEntity<>(response.toArray(), HttpStatus.OK);
}

@GetMapping("/leaderboard")
public @ResponseBody ResponseEntity<?> getLeaderboard() {

    List<Object> response = new ArrayList<>();

    Map<String, Double> overallLeaderboard = LdoD.getInstance().getOverallLeaderboard();
    List<String> users = overallLeaderboard.entrySet().stream()
        .sorted(Collections.reverseOrder(Map.Entry.comparingByValue()))
        .map(Map.Entry::getKey)
        .collect(Collectors.toList());

    List<Double> scores = overallLeaderboard.entrySet().stream()
        .sorted(Collections.reverseOrder(Map.Entry.comparingByValue()))
        .map(Map.Entry::getValue)
        .collect(Collectors.toList());

    response.add(users);
    response.add(scores);

    return new ResponseEntity<>(response.toArray(), HttpStatus.OK);
}

// ------------- WebSocket Methods ------------- //

@MessageMapping("/\{gameId\}/connect")
public @ResponseBody void handleConnect(@Payload Map<String, String> payload) {
    String userId = payload.get("userId");
    String gameId = payload.get("gameId");

    createGameParticipant(gameId, userId);
}

@MessageMapping("/\{gameId\}/tags")
@SendTo("/topic/ldod-game/{gameId}/tags")
public @ResponseBody void handleTags(@Payload Map<String, String> payload) {
    String gameId = payload.get("gameId");
    
    createGameParticipant(gameId, userId);
}
String gameId = payload.get("gameId");
String authorId = payload.get("authorId");
String tag = payload.get("msg");
Object number = payload.get("paragraph");
int finalNumber = Integer.parseInt((String) number);
changeGameState(gameId, ClassificationGame.ClassificationGameState.TAGGING);
saveTag(gameId, authorId, tag, finalNumber);

this.broker.convertAndSend("/topic/ldod-game/"+ gameId + "/tags", payload.values());
}

@Atomic(mode = TxMode.READ)
@MessageMapping("/{gameId}/votes")
@SendTo("/topic/ldod-game/{gameId}/votes")
public @ResponseBody void handleVotes(@Payload Map<String, String> payload) {
    // logger.debug("handleVotes keys: {}, values: {}", payload.keySet(),
    // payload.values());
    String gameId = payload.get("gameId");
    String voterId = payload.get("voterId");
    String tagMsg = payload.get("msg");
    Object vote = payload.get("vote");
    double finalVote = Double.parseDouble((String) vote);
    Object number = payload.get("paragraph");
    int finalNumber = Integer.parseInt((String) number);
    double v = 0;
    changeGameState(gameId, ClassificationGame.ClassificationGameState.VOTING);
    v = saveVote(gameId, voterId, tagMsg, finalVote, finalNumber);
    payload.remove("paragraph");
    String currentTopTag = getCurrentTagWinner(gameId);
    String currentWinner = getCurrentParticipantWinner(gameId);
    payload.put("vote", String.valueOf(v));
    payload.put("top", currentTopTag);
    payload.put("winner", currentWinner);
    this.broker.convertAndSend("/topic/ldod-game/"+ gameId + "/votes", payload.values());
}

@Atomic(mode = TxMode.READ)
@MessageMapping("/{gameId}/review")
@SendTo("/topic/ldod-game/{gameId}/review")
public @ResponseBody void handleReview(@Payload Map<String, String> payload) {
    String gameId = payload.get("gameId");
    Object limit = payload.get("limit");
    int numberOfParagraphs = Integer.parseInt((String) limit);
    numberOfParagraphs = numberOfParagraphs == 1 ? 2 : numberOfParagraphs;
    changeGameState(gameId, ClassificationGame.ClassificationGameState.REVIEWING);
    Set<String> topTags = getTopTags(gameId, numberOfParagraphs);
    List<Map<String, String>> response = new ArrayList<>();
    for (String topTag : topTags) {
        Map<String, String> map = new LinkedHashMap<>();
        map.put("tag", topTag);
        map.put("vote", String.valueOf(0));
        response.add(map);
    }
    String currentTopTag = getCurrentTagWinner(gameId);
    String currentWinner = getCurrentParticipantWinner(gameId);
    payload.put("top", currentTopTag);
    payload.put("winner", currentWinner);
    Map<String, String> map = new LinkedHashMap<>();
    map.put(currentWinner, currentTopTag);
    response.add(map);
    this.broker.convertAndSend("/topic/ldod-game/" + gameId + "\review", response);
}

@MessageMapping("/{gameId}/sync")
public void syncGame(@Payload Map<String, String> payload) {
    String gameId = payload.get("gameId");
    changeToSync(gameId);
}

// ------------- DB Transactions ------------- //

@Atomic(mode = TxMode.WRITE)
private void createGameParticipant(String gameId, String userId) {
}
ClassificationGame game = FenixFramework.getDomainObject(gameId);
game.addParticipant(userId);
}

@Atomic(mode = TxMode.WRITE)
private void saveTag(String gameId, String userId, String tag, int number) {
    ClassificationGame game = FenixFramework.getDomainObject(gameId);
    ClassificationGameParticipant participant = game.getParticipant(userId);
    ClassificationGameRound round = new ClassificationGameRound();

    round.setNumber(number);
    round.setRound(1);
    round.setTag(tag);
    round.setVote(1);
    round.setClassificationGameParticipant(participant);
    round.setTime(DateTime.now());
    participant.setScore(participant.getScore() + ClassificationGame.SUBMIT_TAG);
}

@Atomic(mode = TxMode.WRITE)
private double saveVote(String gameId, String userId, String tag, double vote, int paragraph) {
    ClassificationGame game = FenixFramework.getDomainObject(gameId);
    ClassificationGameParticipant participant = game.getParticipant(userId);
    ClassificationGameRound round = new ClassificationGameRound();

    round.setNumber(paragraph);
    if (getGameState(gameId).equals(ClassificationGame.ClassificationGameState.REVIEWING)) {
        round.setRound(ClassificationGame.VOTING_FINAL_ROUND);
        round.setVote(game.getVotesForTagInFinalRound(tag) + vote);
    } else if (getGameState(gameId).equals(ClassificationGame.ClassificationGameState.VOTING)) {
        round.setRound(ClassificationGame.VOTING_PARAGRAPH_ROUND);
        round.setVote(game.getVotesForTagInParagraph(tag, paragraph) + vote);
    }

    round.setTag(tag);

    // Vote changed in review, makes participant receive a -1 penalty
    if (vote < 0) {
        participant.setScore(participant.getScore() + ClassificationGame.VOTE_CHANGE);
    }

    round.setClassificationGameParticipant(participant);
    round.setTime(DateTime.now());
return round.getVote();
}

@Atomic(mode = TxMode.WRITE)
private void changeToSync(String gameId) {
    ClassificationGame game = FenixFramework.getDomainObject(gameId);
    game.setSync(true);
}

@Atomic(mode = TxMode.WRITE)
private void changeGameState(String gameId, ClassificationGame.ClassificationGameState state) {
    ClassificationGame game = FenixFramework.getDomainObject(gameId);
    if (game.getState().equals(ClassificationGame.ClassificationGameState.REVIEWING)) {
        return;
    }
    game.setState(state);
}

@Atomic(mode = TxMode.READ)
private Set<String> getTopTags(String gameId, int numberOfParagraphs) {
    ClassificationGame game = FenixFramework.getDomainObject(gameId);
    return game.getCurrentTopTags(numberOfParagraphs);
}

@Atomic(mode = TxMode.READ)
private String getCurrentTagWinner(String gameId) {
    ClassificationGame game = FenixFramework.getDomainObject(gameId);
    return game.getCurrentTagWinner();
}

@Atomic(mode = TxMode.READ)
private String getCurrentParticipantWinner(String gameId) {
    ClassificationGame game = FenixFramework.getDomainObject(gameId);
    return game.getCurrentParticipantWinner().getPlayer().getUser().getUsername();
}

@Atomic(mode = TxMode.READ)
private ClassificationGame.ClassificationGameState getGameState(String gameId) {
    ClassificationGame game = FenixFramework.getDomainObject(gameId);
    return game.getState();
}
Listing B.2: ClassificationGame.java

```java
public class ClassificationGame extends ClassificationGame_Base {
    private static Logger logger = LoggerFactory.getLogger(ClassificationGame.class);

    public enum ClassificationGameState {
        CREATED, OPEN, STARTED, TAGGING, VOTING, REVIEWING, ABORTED, FINISHED
    }

    public static final double SUBMIT_TAG = 1;
    public static final double VOTE_CHANGE = -1;
    public static final double SUBMITTER_IS_ROUND_WINNER = 5;
    public static final double SUBMITTER_IS_GAME_WINNER = 5;
    public static final double VOTED_IN_ROUND_WINNER = 2;
    public static final double VOTED_IN_GAME_WINNER = 5;

    public static final int SUBMISSION_ROUND = 1;
    public static final int VOTING_PARAGRAPH_ROUND = 2;
    public static final int VOTING_FINAL_ROUND = 3;

    public ClassificationGame(VirtualEdition virtualEdition, String description, DateTime date,
                               VirtualEditionInter inter, LdoDUser user) {
        if (!virtualEdition.getTaxonomy().getOpenVocabulary()) {
            throw new LdoDException("Cannot create game due to close vocabulary");
        }

        setState(ClassificationGameState.CREATED);
        setDescription(description);
        setDateTime(date);
        setVirtualEditionInter(inter);
        setResponsible(user);
        setVirtualEdition(virtualEdition);
    }

    @Atomic(mode = TxMode.WRITE)
    public void remove() {
        setVirtualEdition(null);

        Tag tag = getTag();
        if (tag != null) {
            setTag(null);
            tag.remove();
        }
    }
}
```
```java
setVirtualEditionInter(null);
setResponsible(null);

getClassificationGameParticipantSet().stream().forEach(p -> p.remove());
deleteDomainObject();
}

public boolean getOpenAnnotation() {
    return getVirtualEdition().getTaxonomy().getOpenAnnotation();
}

public boolean isActive() {
    return DateTime.now().isBefore(getDateTime().plusSeconds(55));
}

@Atomic(mode = TxMode.WRITE)
public void addParticipant(String username) {
    LdoDUser user = LdoD.getInstance().getUser(username);

    if (!getOpenAnnotation() && !getVirtualEdition().getActiveMemberSet().stream().anyMatch(m -> m.getUser() == user)) {
        new LdoDException("User not allowed to play this game.");
    }

    new ClassificationGameParticipant(this, user);
}

@Atomic(mode = TxMode.WRITE)
public void finish() {
    if (getState().equals(ClassificationGameState.FINISHED)) {
        // CANT RECALCULATE OR FINISH A GAME AFTER GAME FINISHED
        return;
    }
    ClassificationGameParticipant participant = calculateParticipantsScores();
    LdoDUser winner = participant.getPlayer().getUser();
    String tagName = getCurrentTagWinner();

    getClassificationGameParticipantSet().stream().filter(p -> p.getPlayer().getUser() == winner).findFirst().get().setWinner(true);
}```
Tag tag = getVirtualEdition().getTaxonomy().createTag(getVirtualEditionInter(), tagName, null, winner);

setState(ClassificationGameState.FINISHED);

public boolean canBeRemoved() {
    return getTag() == null;
}

public Map<String, Double> getLeaderboard() {
    List<ClassificationGameParticipant> participants = getClassificationGameParticipantSet().stream()
        .sorted(Comparator.comparing(ClassificationGameParticipant::getScore))
        .collect(Collectors.toList());

    Map<String, Double> result = new HashMap<>();
    for (ClassificationGameParticipant participant : participants) {
        if (participant.getPlayer() != null) {
            result.put(participant.getPlayer().getUser().getUsername(), participant.getScore());
        }
    }

    return result;
}

public ClassificationGameParticipant getParticipant(String username) {
    return getClassificationGameParticipantSet().stream()
        .filter(p -> p.getPlayer().getUser().getUsername().equals(username))
        .findFirst().orElse(null);
}

public Stream<ClassificationGameRound> getAllRounds() {
    return getClassificationGameParticipantSet().stream()
        .flatMap(p -> p.getClassificationGameRoundSet().stream());
}

public String getCurrentTagWinner() {
    return getAllRounds().max(Comparator.comparing(ClassificationGameRound::getVote)).map(r -> r.getTag())
public Set<String> getCurrentTopTags(int numberOfParagraphs) {
    return IntStream.range(0, numberOfParagraphs)
        .mapToObj(p -> getRoundWinnerTag4Paragraph(p).getTag()).distinct()
        .collect(Collectors.toSet());
}

public ClassificationGameParticipant getCurrentParticipantWinner() {
    String currentTagWinner = getCurrentTagWinner();
    return getAllRounds().filter(round -> round.getTag().equals(currentTagWinner))
        .sorted(Comparator.comparing(o -> o.getTime().getMillis()))
        .findFirst()
        .map(r -> r.getClassificationGameParticipant())
        .orElse(null);
}

private ClassificationGameParticipant calculateParticipantsScores() {
    String tagWinner = getCurrentTagWinner();
    Set<ClassificationGameRound> roundWinners = getRoundWinners(tagWinner, SUBMISSION_ROUND)
        .collect(Collectors.toSet());

    // Submission round

    // Participant that has submitted GAME winner tag FIRST than anyone receives +10
    ClassificationGameParticipant gameWinner = roundWinners.stream()
        .min(Comparator.comparing(r -> r.getTime().getMillis()))
        .map(r -> r.getClassificationGameParticipant())
        .orElse(null);
    gameWinner.setScore(gameWinner.getScore() + SUBMITTER_IS_GAME_WINNER);

    // Participant that has submitted GAME winner tag but is NOT the first equals
    // voter receives +5
    roundWinners.stream().map(r -> r.getClassificationGameParticipant())
        .forEach(p -> p.setScore(p.getScore() + SUBMITTED_LIKE_GAME_WINNER));

    // Paragraph submission and voting
    int numberOfParagraphs = getAllRounds()
        .sorted(Comparator.comparing(ClassificationGameRound::getNumber)
            .reversed())
        .map(r -> r.getNumber())


for (int paragraph = 0; paragraph < numberOfParagraphs; paragraph++) {
    ClassificationGameRound winnerRoundForParagraph = getRoundWinnerTag4Paragraph(paragraph);
    if (winnerRoundForParagraph != null) {
        // Participant that has submitted ROUND winner tag receives 5
        getRoundWinnersSubmit4Paragraph(winnerRoundForParagraph.getTag(), paragraph)
            .map(r -> r.getClassificationGameParticipant())
            .forEach(p -> p.setScore(p.getScore() + SUBMITTER_IS_ROUND_WINNER));

        // Participants that voted in paragraph winner tag receives + 2
        getRoundWinnersVoting4Paragraph(winnerRoundForParagraph.getTag(), paragraph)
            .map(r -> r.getClassificationGameParticipant())
            .forEach(p -> p.setScore(p.getScore() + VOTED_IN_ROUND_WINNER));
    }
}

// ------------- Round 3 ------------- //
// Voted on game winner tag + 5
getRoundWinners(tagWinner, VOTING_FINAL_ROUND).map(r -> r.getClassificationGameParticipant())
    .forEach(p -> p.setScore(p.getScore() + VOTED_IN_GAME_WINNER));

// Impossible to have below zero so every participant gets 1 point
getClassificationGameParticipantSet()
    .stream()
    .forEach(p -> p.setScore(p.getScore() < 0 ? 1 : p.getScore()));

return gameWinner;

private Stream<ClassificationGameRound> getRoundWinnersSubmit4Paragraph(String tagWinner, int paragraph) {
    return getAllRounds()
        .filter(r -> r.getTag().equals(tagWinner) && r.getNumber() == paragraph
            && r.getRound() == SUBMISSION_ROUND);
}
private Stream<ClassificationGameRound> getRoundWinnersVoting4Paragraph(String tagWinner, int paragraph) {
    return getAllRounds().filter(r -> r.getTag().equals(tagWinner) && r.getNumber() == paragraph && r.getRound() == VOTING_PARAGRAPH_ROUND);
}

private ClassificationGameRound getRoundWinnerTag4Paragraph(int paragraph) {
    return getAllRounds().filter(r -> r.getNumber() == paragraph && r.getRound() == VOTING_PARAGRAPH_ROUND)
        .sorted(Comparator.comparing(ClassificationGameRound::getVote).reversed().thenComparing(Comparator.comparing(o -> o.getTime().getMillis())))
        .findFirst().orElse(null);
}

public double getVotesForTagInFinalRound(String tag) {
    return getAllRounds().filter(r -> r.getRound() == VOTING_FINAL_ROUND && r.getTag().equals(tag))
        .sorted((r1, r2) -> r2.getTime().compareTo(r1.getTime()))
        .map(r -> r.getVote()).findFirst().orElse(0.0);
}

public double getVotesForTagInParagraph(String tag, int paragraph) {
    return getAllRounds().filter(r -> r.getNumber() == paragraph && r.getRound() == VOTING_PARAGRAPH_ROUND && r.getTag().equals(tag))
        .sorted((r1, r2) -> r2.getTime().compareTo(r1.getTime()))
        .map(r -> r.getVote()).findFirst().orElse(0.0);
}

Listing B.3: GameRunner.java
private Set<String> playersInGame;

@Autowired
private SimpMessagingTemplate broker;

@Autowired
public GameRunner(final SimpMessagingTemplate broker) {
    this.broker = broker;
}

public void setGameId(String gameId) {
    this.gameId = gameId;
}

@Override
public void run() {
    if (canOpenGame()) {
        while (Instant.now().isBefore(startTime.plusMinutes(1))) {
            try {
                Thread.sleep(600);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
        if (canGameStart()) {
            logger.debug("running game ", this.gameId);
            startGame();
            while (!hasGameEnded()) {
                if (canGameContinue()) {
                    try {
                        Thread.sleep(600);
                        continueGame();
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                }
                else {
                    abortGame();
                }
            }
        }
    }
    @Atomic(mode = TxMode.WRITE)
private synchronized boolean canOpenGame() {
    ClassificationGame game = FenixFramework.getDomainObject(this.gameId);
    logger.debug("running game ", game);
    this.startTime = game.getDateTime();
    if (game.getState().equals(ClassificationGame.ClassificationGameState.CREATED)) {
        game.setState(ClassificationGame.ClassificationGameState.OPEN);
        return true;
    }
    return false;
}

@Atomic(mode = TxMode.READ)
private boolean canGameStart(){
    ClassificationGame game = FenixFramework.getDomainObject(this.gameId);
    // TODO: inform if the game cannot start
    logger.debug("canGameStart size: ", game.getClassificationGameParticipantSet().size());
    return game.getClassificationGameParticipantSet().size() >= 2;
}

@Atomic(mode = TxMode.WRITE)
private void startGame() {
    ClassificationGame game = FenixFramework.getDomainObject(this.gameId);
    game.setState(ClassificationGame.ClassificationGameState.STARTED);
    playersInGame = game.getClassificationGameParticipantSet().stream().map(p -> p.getPlayer().getUser().getUsername()).collect(Collectors.toSet());
    Map<String, String> payload = new LinkedHashMap<>();
    payload.put("currentUsers", String.valueOf(playersInGame.size()));
    payload.put("command", "ready");
    broker.convertAndSend("/topic/ldod-game/", payload.values());
}

@Atomic(mode = TxMode.WRITE)
private void abortGame() {
    ClassificationGame game = FenixFramework.getDomainObject(this.gameId);
    game.setState(ClassificationGame.ClassificationGameState.ABORTED);
    Map<String, String> payload = new LinkedHashMap<>();
    payload.put("currentUsers", String.valueOf(0));
    payload.put("command", "aborted");
    broker.convertAndSend("/topic/ldod-game/", payload.values());
}
private boolean canGameContinue() {
    ClassificationGame game = FenixFramework.getDomainObject(this.gameId);
    return game.getSync();
}

@Atomic(mode = TxMode.WRITE)
private void continueGame() {
    Map<String, String> payload = new LinkedHashMap<>();
    payload.put("command", "continue");
    ClassificationGame game = FenixFramework.getDomainObject(this.gameId);
    game.setSync(false);
    broker.convertAndSend("/topic/ldod-game/" + this.gameId + "/sync", payload.values());
}

@Atomic(mode = TxMode.READ)
private boolean hasGameEnded() {
    ClassificationGame game = FenixFramework.getDomainObject(this.gameId);
    return game.getState().equals(ClassificationGame.ClassificationGameState.FINISHED);
}
Game screens
Figure C.1: Game starting screen.

Figure C.2: Not enough users joined to play screen.

Figure C.3: Game design sample.

Figure C.4: Leaderboard page.
Figure C.5: About page.

Figure C.6: Feedback page.

(a) Notification with success message.  
(b) Notification with failure message.  
(c) Notification with an information message.  
(d) Error page.

Figure C.7: Different game feedback provided to players.
Questionnaire
Figure D.1: Questionnaire: first section.
<table>
<thead>
<tr>
<th>Caracterização do jogo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifique quanto o jogo quanto à dificuldade de entendimento do mesmo (objetivo e regras). *</td>
</tr>
<tr>
<td>Muito FÁCIL de comprender</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>Nota divertido.</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>Muito divertido.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Consideraria jogar novamente? *</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sim.</td>
</tr>
<tr>
<td>Talvez.</td>
</tr>
<tr>
<td>Não.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Recomendaria o jogo a outra pessoa? *</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sim.</td>
</tr>
<tr>
<td>Talvez.</td>
</tr>
<tr>
<td>Não.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>O jogo fez com que fosse interessante ler e saber mais sobre o Livro do Desassossego e Famando Pessoas em geral? *</th>
</tr>
</thead>
<tbody>
<tr>
<td>Não, nada.</td>
</tr>
<tr>
<td>Um pouco.</td>
</tr>
<tr>
<td>Sim, bastante.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>O que é que gostou mais no jogo?</th>
</tr>
</thead>
<tbody>
<tr>
<td>A sua resposta</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>O que é que gostou menos no jogo?</th>
</tr>
</thead>
<tbody>
<tr>
<td>A sua resposta</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Que melhorias acha que podem ser feitas no jogo?</th>
</tr>
</thead>
<tbody>
<tr>
<td>A sua resposta</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Caso tenha alguma sugestão para o jogo não hesite em escrevê-la. As suas ideias são muito importantes!</th>
</tr>
</thead>
<tbody>
<tr>
<td>A sua resposta</td>
</tr>
</tbody>
</table>

---

**Figure D.2: Questionnaire: second section.**
Figure D.3: Questionnaire: answers-1.

Figure D.4: Questionnaire: answers-2.

Figure D.5: Questionnaire: answers-3.
Figure D.6: Questionnaire: answers-4.

Figure D.7: Questionnaire: answers-5.
Figure D.8: Questionnaire: answers-6.