INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Praxis Market Gap Analysis

JOÃO PEDRO MOREIRA ALVES DIAS
Outubro de 2016

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Praxis Market Gap Analysis

João Pedro Moreira Alves Dias

Dissertation in Computer Science (Computer Engineering)

Graphics Systems and Multimedia

Mentors:

Nuno Filipe Fonseca Vasconcelos Escudeiro
Paula Maria de Sá Oliveira Escudeiro

Porto, October 2016

Acknowledgement

I would like to express my sincere gratitude to my supervisor Nuno Escudeiro for his time dedicated to my dissertation, his continuous support and suggestions.

I would like to thank also to my co-supervisor Paula Escudeiro for her help and useful suggestions and all the members of Praxis.

Lastly, I would also like to express my gratitude to my beloved family, without their support I could not have finished my master's studies. Finally, my thanks go to my girlfriend for her continuous support.

You will always be remembered **Drina**.

Resumo

O Praxis é um ponto de encontro entre estudantes do ensino superior e empregadores num mercado de trabalho globalizado. Os principais fatores impulsionadores do mercado são as ofertas de estágio, submetidas pelos empregadores, e as palavras-chave das pesquisas realizadas pelos estudantes que se encontram à procura de um estágio. A preocupação principal do Praxis está relacionada com a discrepância que possa existir entre os estágios disponíveis e as cerca de 1000 pesquisas realizadas por dia. Este trabalho visa preencher eventuais lacunas existentes no mercado Praxis entre a oferta – propostas de estágio -- e a procura – palavras chave usadas nas pesquisas.

Este trabalho analisa o resultado da aplicação de diferentes técnicas de processamento automático de texto em combinação com documentos orientados a dados. Esta associação tira partido de todos os recursos existentes nos navegadores de internet mais modernos, proporcionando assim uma experiência mais agradável para o utilizador, incluindo unicamente as informações que são relevantes e que, de outra forma, passariam despercebidas para o utilizador.

Além disso, as técnicas de processamento automático de texto servem para analisar as ofertas de estágio e as palavras-chave pesquisadas, sendo que podem melhorar a correspondência entre a oferta e a procura no mercado virtual Praxis.

O trabalho Praxis Market Gap Analysis demonstra também o potencial de usar nuvens de etiquetas e técnicas de clustering em combinação com documentos orientados a dados. Esta abordagem contribui para uma melhoria das representações gráficas baseadas em dados e da qualidade da informação disponibilizada ao utilizador final.

Os principais resultados deste trabalho sugerem que os diferentes algoritmos de clustering têm resultados distintos no que respeita à organização do corpus, sendo que isto afeta a eficácia do processo de clustering. Os resultados da avaliação orientaram-nos para os algoritmos de clustering mais eficientes atendendo ao contexto e às necessidades do Praxis.

Palavras-chave: Processamento automático de texto, Nuvem de etiquetas, Clustering, Documentos orientados a dados

Abstract

Praxis is a meeting point where higher education students meet employers in the global labor market. The main entities driving the market are the internship offers, submitted by the providers, and the keyword searches from the students looking for an internship. The core concern of the Praxis network is related to the mismatch between the internships being offered (supply) and the searches being requested (demand). Hence, this work aims to fill the gap in the Praxis market between the demand and supply.

This work analyses the outcome from the application of different text mining techniques in combination with Data-Driven Documents in the Praxis network. Such combination takes advantage of the full capabilities in modern browsers and thus provides a pleasant user experience by displaying only the relevant information that was previously unseen but contained in the data.

Furthermore, using text mining techniques to analyze the internship offers and the keyword searches being made in the database might improve the matching between demand and supply in the Praxis virtual market.

The Praxis Market Gap Analysis portraits the potential of using tag clouds and clustering while combining Data-Driven Documents with mined data. This approach increases the number of data-driven representations and the quality of information that the final user perceives.

The main findings of this study suggest that the different clustering algorithms have disparate results in regard to the organization of the input corpus which impacts the effectiveness of the clustering process. The evaluation results guided us towards the most efficient clustering algorithms for the specific context and needs of Praxis.

Keywords: Text Mining, Tag Clouds, Clustering, Data-Driven Documents

Index

List of images

List of Tables

Acronyms

List of Acronyms

PI	Project/Internship
API	Application Programming Interface
ISEP	Instituto Superior de Engenharia do Porto
XML	eXtensible Markup Language
JSON	JavaScript Object Notation
DOM	Document Object Model
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
REST	Representational State Transfer
SVG	Scalable Vector Graphics
D3	Data-Driven Documents
MVC	Model-View-Controller
UML	Unified Modelling Language
PHP	HyperText Preprocessor
URL	Uniform Resource Locator
HTTP	Hypertext Transfer Protocol
SSH	Secure Shell

1 Introduction

Nowadays, most of the data is present in digital, rather than paper form. (Weiss et al. 2010) As a consequence, a new field in computer science has evolved. Data mining is already a mature technology and suitable for any project where interpretation of data is a key to the success. Its main approach is to find patterns in data that are unseen at first glance, so it seems straightforward to take advantage of it when dealing with large volumes of data.

However, its implementation is still distant from universal, there are already highly advanced techniques that allow data analysis with high level of confidence. Text mining is a good example of such techniques and the main difference when compared to others is that text mining does not expect data to be structured for discovering useful patterns from texts. Another important aspect is that this specific technique of data mining has become even more efficient given the fast paced growth of computing power and its affordable price.

It is said that where this emerging technique is used, both text mining and analytics are successfully applied, in order to achieve new knowledge that surpasses the first and superficially sight about data. (McDonald & Kelly 2015)

This document is related to the dissertation of the second year of the Master's Degree in Informatics Engineering at ISEP and was guided by leader Professor Nuno Escudeiro.

1.1 Praxis network

Praxis is a European platform that handles a virtual market based on project/internship offers. This project has been funded by the European Commission and it is designed to be a consortium of higher education institutions, research labs, companies and other institutions. These institutions can submit their current vacancies for students on the website and mutually students can search and potentially find an internship or project there.

The main advantage of this platform when compared to others is the commitment and partnership within the network where all institutions focus their effort to strengthen and facilitate the project/internship experience.

1.2 Motivation

Regarding the area of data mining being an interdisciplinary subfield of computer science, I never had the opportunity to work or study it, whether in my academic or professional experience. From my point of view, this area has shown the potential of adding huge value to many projects that have been taking advantage of it.

1.3 Problem

This project is about developing a flexible tool directed to the Praxis administrators and its purpose is to promote the matching between Project/Internship (PI) offers and searches made through the Praxis platform.

The website offers a wide range of projects and internships, so it is highly desirable that when a student makes a search, whether with one or multiple keywords expressing his own interest, at least one result is shown. Otherwise it might lead to the user frustration and discouragement for using the platform. Moreover, an offer has ideally as many skilled candidates as possible, so in the case that a vacancy has been a long time without any candidate, it is likely that the host will step backwards against the Praxis network for posting different offers next time.

The main goal is to reduce both, the number of unmatched PI offers and the number of searches returning empty results.

1.4 Hypothesis

The application of text mining techniques on the Praxis website can be used in an efficient manner in order to detect gaps and discrepancies among the available proposals and the user's searches. Moreover, such techniques are automated, therefore it highly decreases the required time for data analysis. Regarding the expenditure of running text mining techniques, it is generally low-priced and competitive as compared to the project's dimension. In addition, text mining techniques can analyze large volumes of data and detect patterns that were, at first instance, unseen.

In conclusion, it is a part of work's supposition, that text mining techniques are endogenously connected to what is the ambition and belief of Praxis, having an answer to each user's single search made on the website. Thereby enhancing the correspondence on proposal's versus user's demand.

1.5 Approach

The supply/demand matching will be based on the analysis of the content and metadata of searches and PI. All of this data is available on the Praxis website's database, therefore several techniques already present in the area of text mining and clustering can be used to strive the current mismatch.

It is a fact that several thousand searches are being made on the website given certain keywords, so if we match and register all of those keywords and compare them to the available offers, we can detect which offers have low or high demand and expose mismatches. This information will be crucial to guide the Praxis partnership in its procurement of new partners.

1.6 Expected results

The expected results of this thesis are to present and show how text mining techniques can potentially help the matching between supply/demand on the Praxis virtual market.

The main goals are to:

1. Analyse the data that is available;
2. Specify the services to provide and the way to provide them;
3. Implement a clustering procedure for supply (internship offers) and demand (search keywords);
4. Implement the procedures to extract, clean, aggregate and analyse the information available at the Praxis database (MySQL);
5. Design and develop the user interface in accordance with the brand guide of Praxis;
6. Design and develop the methodology to generate periodic reports on the supply/demand mismatches and, eventually, suggest actions to correct them;
7. Implement and deploy the Market Gap Analysis service.

1.7 Outline

This section briefly describes each of the seven chapters composing this dissertation.

Chapter 1 introduces the project, its context and motivation.

Chapter 2 presents a literature review on the context of this project, the different text mining techniques and the possible challenges.

Chapter 3 shows a brief research on the existent tools and how we can benefit from them.

Chapter 4 presents the design of the product, showing the functional requirements, the solution architecture and a visual mockup for what it can be the web platform Praxis Market Gap Analysis.

Chapter 5 is where the conceived Design is implemented, forasmuch as it shows the mapping from design to the final solution, whilst always adopting the best practices within software engineering.

Chapter 6 explains the evaluation of the solution, whether by testing its efficiency or analysing the added value to who is using it.

Lastly, **Chapter 7** presents the conclusions learnt by doing this work and its limitations. Future work suggestions related to this dissertation are also presented.

2 Context and state of the art

This chapter presents the context in which the project is involved and an overview of the current text mining techniques that might be useful to this project.

2.1 Contextualization

Mobility exchanges are increasing every year, since more and more young people are using mobility programmes to gain valuable work experience abroad. (European Commission 2014) Therefore the demand on the student's needs when it comes to search for PI, whether within European Union or outside is higher and higher and there is a vast of well-known reasons for that, such as, big support, high budgeting for these types of programmes, great feedback on each ex Erasmus student experience and the low rate of unemployment for people that either studied or were trained abroad.

Thus the added value that is inherited from extensive analysis of the current gap on the Praxis market is tremendous. Despite the different tools that are already in the market and that add huge knowledge about tracking and website traffic, as e.g. Google Analytics, there is still space to grow for both business and knowledge about user's insights.

At the moment there were approximately 300 000 searches (data was gathered on 8th February 2016 from Praxis database) for PI on the Praxis website, and from those 300 000, 37 522 lead to none results, which represents 12.5% of the queries returning empty results. If we create tools that might help us on reducing these numbers, this will be reflected in a direct impact on the user's first impression and satisfaction when using the Praxis platform.

Wrapping it up, it is proposed during the time of this work to explore different techniques, in order to know more about what job offers are available and have either low number or no candidates and which keywords are being used by students and explore the reason why part of those queries are returning unsatisfactory results.

2.1.1 Value proposition

The value proposition in our work consists in reducing the number of offers without any candidate and the number of searches returning empty results. Consequently, increasing and contributing to the general and overall satisfaction of the users using our platform. It is targeted to enhance the experience of both students searching for an offer and entities that are making offers available. Thereby attracting a worldwide variety of students that are interested in using Praxis to get to know which PI are available. It is designed to be used directly by Praxis administrators that are responsible to find the gaps and plan actions to correct them.

2.1.2 Market value

There are assumptions that have been showing the potential of such platforms where Praxis is included, some of the facts are as e.g.:

- 1 in 3 Erasmus Trainees are offered a position by the company they worked for (European Commission 2014);
- Five years after graduation, the unemployment rate of young people is 23% lower when compared to non-mobile peers (European Commission 2014);
- 40% increase in funding when compared to the predecessor programme (European Commission 2014).

It is expected that the need to use website offers like Praxis will increase in the same pace as the pursuit for new work experiences abroad. Thereby, the more advanced will our matching mechanism between users queries and available offers be, the more people will be using our platform, consequently leading to more students seeking for a work experience and more institutions posting new offers.

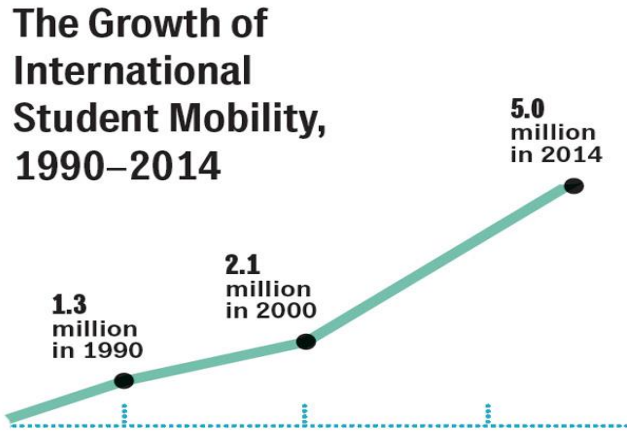

Image 1: Student mobility growth since 1990, adaptation from (ICEF 2015)

As shown in the above Image 1, the market forecasts and anticipates greater demand for vocational training abroad. (ICEF 2015)

The reason for this outstanding growth is strongly associated to our new era, when it is relatively easy to gather and get all the useful information within platforms like Praxis and others. Lastly, if systems like ours are able to predict what the market is expecting from us, we may assume that we have a bright future ahead.

2.1.3 Business Model Canvas

The proposed Business Canvas Model presented in Table 1 states that the key partners are all the institutions/companies that are looking for talented and skilled students that want a job experience abroad. Moreover, higher education institutions using Praxis can bridge the gap between its institutions and companies, that are at the same time attracting foreign students and exposing them to the global market. It is important to underline the value of corresponding to what are the key partner's expectations within the Praxis' platform, since the reason for Praxis' existence is to be the best place where students can find and/or share international internships and projects among the best working places.

The main key activities for this thesis are: the research about how text mining techniques can endogenously enhance the quality of information retrieved from the Praxis market database and implement a modern web tool that makes the use of such techniques and appeals to the user's retention on data. On the other hand, the key resources are the knowledge either in the area of text mining or in the area of modern web frameworks, whilst the project's costs are implicitly related to the hosting of the application, however such costs are and will be supported by a funding from the European Commission based on the Lifelong Learning Programme.

As previously described in the section 2.1.1, the different value propositions related to the project are to promote the matching between PI offers and student's demand, reduce both the number of offers without any candidate and user's searches without any result, provide a better overview and insight about what is going on the Praxis market database taking advantage of two different but complementary areas, such as text mining and web development. Lastly, potentially create new partnerships and strengthen existent ones while the platform clearly responds to what is the intention of the partners.

The customer segments being affected by this business plan are, as previously mentioned, the students that are looking for a reliable job opportunity out of their home countries, the companies looking for talented and skilled candidates thus improving the access to young talents. Finally, the person responsible to find and detect mismatches when using the tool, likely the Praxis administrator.

It is reasonable to mention that, finding a mechanism that detects gaps in the market, will contribute to a higher efficiency in the matching process, therefore contributing to maintenance of the general interest of all the stakeholders.

In conclusion, the Praxis Market Gap Analysis tool promises to be the tool that will find the mismatch in the proliferation of available vacancies and the student's needs, helping the target audience to join international teams and give them the opportunity to choose the PI that they are really interested in.

Table 1: Praxis Market Gap Analysis Tool Business Model Canvas

<table>
<tr>
<td>Key Partners</td>
<td>Key Activities</td>
<td>Value Proposition</td>
<td>Customer Relationships</td>
<td>Customer Segments</td>
</tr>
<tr>
<td>Higher Education Institutions

Research Labs

Companies

Associations

Other Institutions</td>
<td><u>Research</u>

Research about modern web frameworks

Research about data/text mining techniques

<u>Implement</u>

Develop a flexible tool for Praxis administrators

Implement data driven documents

Key Resources

<u>Team</u>

Good knowledge in data mining, text mining techniques and modern web frameworks

Hosting servers</td>
<td><u>Promote matching</u>

Promote matching between PI offers and user searches

<u>Reduce</u>

Reduce the number of unmatched PI offers and searches returning empty results on Praxis

<u>Better overview about Praxis</u>

Data driven docs better allow the manipulation of data

<u>Enhance new partnerships</u>

Such information will be crucial to guide the Praxis partnership in its procurement of new partners</td>
<td><u>Automation</u>

Decreased time for data analysis within Praxis

<u>Responsive</u>

Praxis Market Gap Analysis web tool will be responsive

<u>Fast</u>

The tool shall be fast when reading large sets of data

Channels

<u>Web</u>

Praxis Market Gap Analysis web tool

<u>Web services</u>

Feed about valid offers via WS</td>
<td><u>Praxis admin</u>

Praxis admin will have a web tool that handles the visualization and manipulation of data within Praxis

<u>Students</u>

Students looking for a PI abroad have better chances to find what they are looking for

<u>Companies</u>

Companies looking to disseminate their internships and research projects to a wide audience of BSc, MSc and PhD students</td>
</tr>
<tr>
<td colspan="3">Cost</td>
<td colspan="2">Revenue Streams</td>
</tr>
<tr>
<td colspan="3"><u>Team</u>

Project Manager
Developers

<u>Hosting</u>

Carrot2 Document Cluster Server (Java)
Backend server (Node.js)
Web application (Angular.js)</td>
<td colspan="2">Funding from European Commission

Lifelong Learning Programme</td>
</tr>
</table>

2.2 Relevant corpora

This section is related to which corpus (in text mining corpus refers to a collection of text documents) is to be studied and analysed in the context of Praxis Market Gap Analysis. There are mainly two sources of information that currently feed the website, the offers (supply) and respectively the peremptory users' searches (demand). The goal of our study and analysis is to boost the synchronization between offers and searches. Although at the moment there is a significant amount of searches returning no results, it is our conviction that amidst those searches there are offers awaiting.

The supply is represented as the following:

Table 2 : Normal structure of an offer description in Praxis

Field	**Description**
Title	Offer's title
Description	Description of the offer
Job related skills	What skills are a prerequisite to the offer
Benefits	Benefits of the offer
Host	Usually the institution
City	Location of the offer
Study domain	What disciplines will be part of the PI
Study degree	Required study degree
Languages	Required languages
Valid until	Until when the offer is valid

Whilst the supply is undoubtedly composed by information that is more or less structured, the demand is free of restrictions, since the users may type whatever they want to search. It is the responsibility of the matchmaking mechanism to figure it out which queries have enough relevance to return potentially related offers available. Usually the search is made using either a term or a group of terms that can be represented by the name of a country, city, study domain, language, given the diversity of possibilities, the queries might be quite different, therefore it would be an advantage to know which terms are being used the most in a specific time. This is just an example out of the many choices that this project has, in order to address the problem of ineffective searches.

The main purpose of this work is to facilitate and give answers to what needs the users might have, thereby smoothing the user experience and to find possible but unknown connections between user's questions and offers available.

The supply/demand matching will be based on the precise analysis of the content and metadata of both searches and PI offers, then likely to discover more about what data is available and what are the reasons for an unsuccessful experience within Praxis.

2.3 Text mining

As presented in the first introductory chapter, text mining is a ubiquitous data mining technique associated with the big usage of computers that generate data in digital form. Text mining can be briefly described as a process that involves retrieving information and structure from a large amount of text. It is also a learning process, meaning the more data has been correctly analysed, the better the results will be in the future. (Weiss et al. 2010)

2.3.1 Where can it be applied?

Text mining has many applications, as e.g. it is used by search engines like Google and Bing, so they scan the web page by its content and then present the matches (typically a list of results) with higher rate to the user; it can also be used to mine scientific research papers in order to link and make possible connections that were unexplored before. (Gary 2012)

There are endless applications for text mining, it is not just limited to computer sciences since it can be extremely useful in a different range of sciences, from social science to education and business. (Filippov 2014) As an illustrative example a historian with enough technological skills and provided with a digital archive can interpret what terms were more dominant in a given interval of time. (Hargreaves et al. 2014)

2.3.2 Why text mining?

Regarding the last technological developments, we can easily guess that large amounts of data are mainly represented by text, whether in social media with a post that shares a specific status or a notice of invoice when someone makes a simple transaction online.

Each day we are directly contributing to the exponential growth of data represented in digital form (McMonagle 2013) and therefore leveraging this specific type of data might improve decisions and predictions in many types of businesses. Nevertheless, there are already several mature text mining techniques that can be used to improve the overall efficiency of a search on the Praxis Network platform.

In the last years there has been a significant increase in the number of both publications and patents related to text and data mining worldwide. Academic communities around Europe are aware of the role and importance of data analytics. There were already many academic researches that already used text mining at a very basic level and those who have used are likely to use it again in the future as the knowledge and benefits become more visible. (Filippov 2014)

The proliferation of information through the Internet over the past two decades has increased at very fast pace and the volume of data is duplicating every 2 years. (IBM 2016) It seems to be the time to adapt such techniques in the favour of new insights and wider understanding about data.

In conclusion, the majority of articles referring to text mining are written in English, since it is not only used by native speakers but also by other scientists around the world (Filippov 2014), thus simplifying our process of finding relevant and credible information.

2.4 Text mining techniques

Despite the existence of several techniques related to the area of text mining, we will be describing only those that are suitable to the project.

2.4.1 Information retrieval

Information retrieval is strongly associated with online documents (Weiss et al. 2010) and its process starts when a user queries the system, as e.g. a specific search in web search engine, and then finally gets a set of results (matched documents) for the input. Nowadays, the way how people interact and search for the information has changed radically, thus millions of people are taking advantage of information retrieval, whether for searching on the web or searching on their email.

The fundamental concept is about finding correlation between the user's input and the document collection. The Image 2 illustrates this concept.

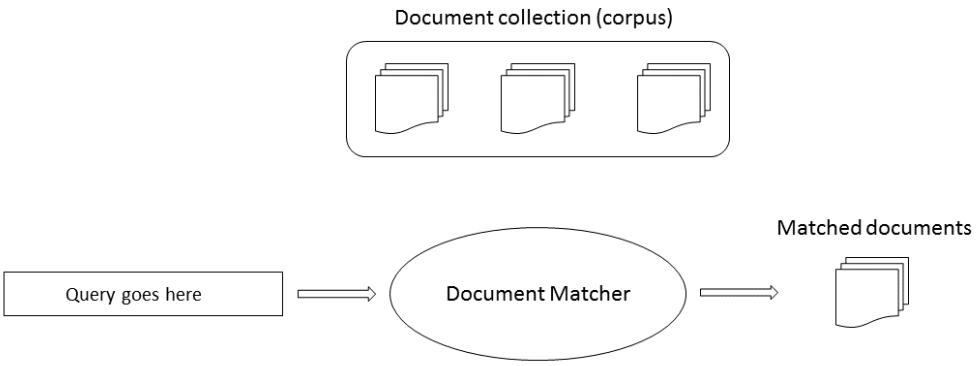

Image 2: Information retrieval, adaptation from (Weiss et al. 2010)

This specific text mining technique is remarkably important to us, since the users on the platform are querying the system with a range of specific keywords. The students are our target and we shall provide them with the ability of finding relevant information about what their queries are. The user might search for a project, internship, countries and also may search in different languages, which causes big ambiguity on the search. It is likely to be one of the first challenges that we need to address, so we can have better efficiency in our Document Matcher.

2.4.2 Information extraction

As previously presented, text mining seeks for patterns in text that is naturally unstructured (Weiss et al. 2010). Information extraction attempts to scan unstructured text and locate specific items that might be structured. It essentially finds a predefined set of concepts in a specific domain, thus leaving out any other irrelevant information. (Piskorski & Yangarber 2013) This technique is not easy to implement given the complexity and general ambiguity of natural language, however improvements have been made into it.

2.4.2.1 Named-entity regocnigtion

This classic task of information extraction aims to solve the problem of identifying possible and predefined types of entities, in regard to the example of Praxis market network, such as, location (PI location), languages (what languages are required to perform the PI), temporal requirements (how long the PI will last). The mentioned task might also include the extraction of descriptive text from text (Piskorski & Yangarber 2013), as e.g., in Praxis offers we have a field called *Job related skills* where it is usually present the required soft-skills, technological skills, so in the end all the different attributes that define the requirements of an ideal candidate for the proposal.

2.4.2.2 Co-reference resolution

Co-reference resolution is the process of determining different words as representing the same entity. For instance: Prague, Praga and Praha - all of these words represent the same entity but they are written in different language, it is likely that if a student is searching for a work in “Prague”, but the job offer has its job location in “Praha”, the result to the query will be empty, which is not likely to be the desired and right behaviour, since both represent the same thing.

2.4.2.3 Tag cloud

Tag clouds are emerging since they are really straightforward and appealing visualization methods for simple text. (Heimerl et al. 2014)

This technique is normally used to feature the most used words in a text. In regard to the context of our Praxis Market Gap Analysis we shall exploit:

- The keywords most associated to empty result sets;
- The keywords most associated to a complete list of results;
- The keywords that are often present in valid offers;
- The keywords that are often present in searches.

This technique should be considered as a must-have since it has an understandable interface, it is also more visually engaging than a table data and it is simple and clear. A tag cloud can be compared to a traditional histogram (graphical representation that displays the frequency of a relatively short number of items), whereas the tag cloud is usually able to display a larger

number of items and with the advantage of being possible to insert a hyperlinks on each item, therefore allowing the interaction of the user with it, such feature is lacking on histograms. (Kaser & Lemire 2007)

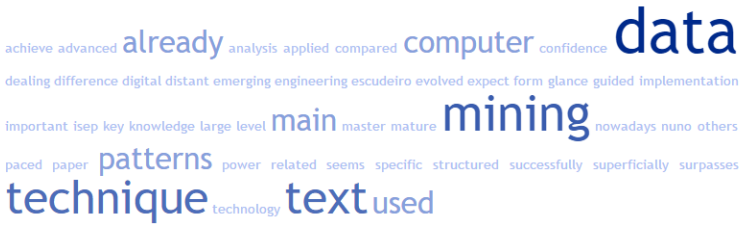

Image 3: Tag cloud about the word frequency in our introduction

The Image 3 shows exactly how simple and valuable in terms of information a tag cloud might be. The words that occur the most within our introduction are depicted and highlighted.

The algorithm for making a tag cloud is pretty straightforward. It computes word's frequency in the corpus and represents them with a font size that is proportional to each term frequency. The tricky part is to guarantee that the words visually displayed do not intersect.

2.4.3 Information retrieval versus information extraction

Information retrieval and Information extraction are often mixed-up, the task of the former is to select a range of results from a document collection ordered by its level of relevance in regard to a particular query, whilst the latter does not deal with any selection or ranking a list of documents, its goal is to extract from those documents the facts about specific types of entities, events, relationships, thus leading to a richer representation of what is in fact present in a text.

2.4.4 Document clustering

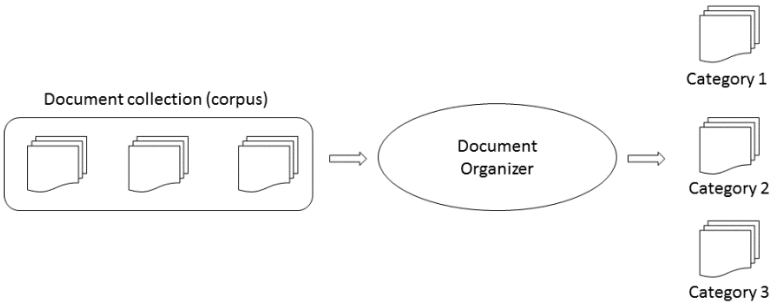

Image 4: Document clustering, adaptation from (Weiss et al. 2010)

The technique of document clustering involves a collection of documents where its goal is to divide those documents into a set of folders that share similar characteristics. Briefly said,

document clustering marks identical documents into the same cluster. There are already plenty different algorithms for clustering analysis and we will explore those that might benefit us.

The Image 4 mirrors how documents with unknown structure are put to categories. In reference to the Praxis network, if we study what keywords might characterize a cluster, we can learn more about what offers are likely to have lower or higher demand.

2.4.4.1 Clustering algorithms

There are already different clustering algorithms that we can adapt to our work. The main three algorithms available are:

- **K-means**, it is one of the most used algorithms for the typical clustering. It is easy to handle and the results are improved by repetition. It searches randomly for its closest neighbours and this last detail might be a key factor to exclude it. (Hartigan & Wong 1979)

- **Lingo**, it is based on singular value decomposition, as k-means, it will also search for its closest neighbours, however wisely adapts a factorization matrix. (Law Osiski et al. n.d.)

- **STC**, stands for suffix tree clustering, and it firstly scans the entire text present in the document, rather than starting to search first for its neighbours. (Srinivasarao et al. 2012)

The empirical experimentation is a decisive step in order to know what algorithms work the best for a given corpus. This evaluation will be based on the number of clusters created and on how many documents do not fall to the *unknown* category.

At this early stage it is neither expected to know already which algorithms can be the best fit to our needs nor do we have the capability to know without experimenting and evaluating them in practice.

2.5 Data-Driven Documents

In data graphics, the appropriate and well-designed animated transition notably improves the graphical perception of both syntactic and semantic levels of analysis. (Heer & G. Robertson 2007) Firstly, this is justified by the intuitive and engaging nature of animation to the user, secondly peripheral vision easier detects motion in a dynamic document. On the contrary static documents can be boring and more difficult to perceive. Data-Driven Documents (D3) try to fulfil this gap in the user's retention, thereby enhancing the common inspection of data and an improved manipulation of a native graphical representation in the Document Object Model. (Michael et al. 2011) D3 is carried by notable features as performance and interoperability, thus meaning speed in rendering and cross-browser compatibility whereas it has been widely used by researchers, journalists and business people for presenting and sharing their data.

The main purpose of Data-Driven Documents is to render a better and more intuitive navigation to the user who is looking for a different approach when it comes to read and interact with data.

Therefore, the efficient manipulation of documents based on data are the crux of the problem that Data-Driven Documents aim to solve.

Finally, unlike other graphic frameworks, the features are built and are coming directly from web standards as e.g. HTML, SVG and CSS (David & Clarence 2015) what makes easier to debug and covert data that is at first in lightweight data-interchange format into beautiful graphical representations.

2.6 Combining text mining with Data-Driven Documents

The combination of text mining with Data-Driven Documents suggests to be inevitable according to the purpose of this work. The former takes advantage of the algorithms and techniques within the area so it can retrieve meaning from a big set of unstructured text, while the latter makes use of the latest technologies in web browsers to show the information with greater detail and wider options of manipulation to the user.

In regard to the combination of both technologies, it is important to have a sequence of tasks where the first is to know what structure of data will be analysed, secondly what analytical methods will be applied therefore generating insight and finally how the findings might be presented. If the purpose is to provide a mechanism of interaction to the text mining results, this would be the best approach to the user. D3 uses data-driven transformations and transitions within CSS3 what turns the data to be interactive. (D3.js 2015)

As previously described, when a person goes to read a newspaper or visit a newspaper's website, he normally faces numerous types of data visualizations and there is a good reason for that. These technologies can be used together in order to augment the user's ability when reading data and it is known that a good data visualization might turn hours of careful study into a quick insight. Moreover, visualization is the key in helping the user to discover new patterns and trends within data, while also promoting the gain of knowledge and insight into the patterns discovered by the used algorithms.

There are already many text mining visualizations that can be used, a good example are multidimensional data graphs, as e.g. Pie chart, Point chart, Column chart, etc. These multidimensional graphs allow users to visually compare data dimensions with other data dimensions adopting the spatial coordinate system (typically x-axis versus x-axis). (Soukup & Davidson 2002) All of these typical bi-dimensional visualizations are already available on D3.js [1] library, what facilitates the process of joining the text mining results with the Data-Driven Documents.

The use of data visualization tools within D3.js is remarkable important within this project, and it can be justified to the way of the human brain processes information, since for the brain it is much easier to visualize large sets of data using e.g. a chart rather than looking to a spreadsheet.

[1] https://github.com/d3/d3/wiki/Gallery (visited on 10/02/2016).

The advantages of using data visualization are related to the speed of comprehending the information much quicker and the potential of identifying relations and patterns.(SAS 2015)

Finally, it is decisive to understand what data we will provide, so we can better know what different types of data visualization tools we need to use, in order to bring meaning and sense to what was unseen before.

2.7 Related work

The purpose of this section is to explore what related work is ready to be used and might be beneficial to us, either in the case of clustering or in the case of providing a smooth information retrieval experience. With regard to the former there are already several online tools available that cluster a range of words into different categories.

Furthermore, there are already plenty of different online applications that do exactly what is our main goal and it is our intention to merely show how relevant those applications might be to the Praxis Market Gap Analysis, since the possibility to learn from others and the will to improve is desired for us.

Nonetheless, in the end it is important to depict the reason why these applications are not sufficient for the work's purpose and why there is the need to develop something new and adapted to Praxis' problem.

2.7.1 Lucene

Lucene is a free and open-source information retrieval software library that is entirely written in Java (Apache Lucene 2011) and this API is used by Praxis website as the main engine for the searching capability since it has been widely recognized for its advantage on single-site searching. (McCandless et al. 2010)

A great example of its application is the real time search on the social network Twitter (Siegler 2010) and a great advantage of using it is the portability to different programming languages as, e.g. C++, PHP (used in Praxis), Perl and others. (Apache Lucene 2012)

2.7.2 Carrot2

Carrot2 is an open source search result clustering engine (Carrot2 2002) and it is used for organizing sets of documents into different categories, thus giving an overview what each document contains. This tool like Lucene is also implemented in Java, but there are other different types of native APIs such as PHP and C# (Carrot2 2002) that have the ability to call a REST interface.

This open source API can be potentially useful to us since we want to make a collection of our own documents, thereby manipulating the REST interface provided by Carrot2.

The odds of creating our own application using this technology are pretty high, since we should not reinvent the wheel.

2.7.3 VocabGrabber

This web application analyses any text that is pasted into it, generating a list of the most useful vocabulary words and presents how those words are used in the given context. It also creates a word cloud according to the frequency of the words and displays the definition and examples of words that are used in the same context.

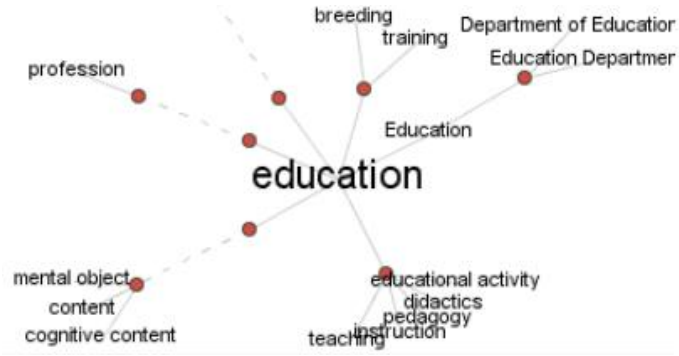

Image 5: Graphic view over the education word on VocabGrabber[2]

The idea of using it as pre-reading tool can be powerful, thus it is one of our main objectives to have a similar interface in our project.

2.7.4 Wordle

Wordle is a web tool where the user can paste text in order to generate word clouds. Afterwards, as a tag cloud, the words that appear the most frequent are easier to detect given their size within the cloud. It is also possible to change the world clouds to have different text fonts, layouts and colour schemes. (Feinberg 2014)

Besides the utility and potential of the tool for our context, the Wordle's source code and all its rights are reserved to IMB Corporation since the author wrote the core layout algorithms during on company time. It is obvious that this represents a clear limitation for further development within Praxis' project using this web application.

[2] Retrieved from https://www.visualthesaurus.com/vocabgrabber/ when searching for "Education" (visited on 15/02/2016).

In regard to the mentioned limitation, the author published some parts of Worlde's source code as open-source, albeit only dealing with breaking text into words and the recognition of similar words.

2.8 Expected challenges

It is important at this early stage to be aware of what challenges we have ahead and what restrictions can we have, so we can better prevent possible changes according to our plans.

Regarding the topic of challenges, the first will be mainly associated to the choosing of what text mining techniques will be definitely applied and in what circumstances. Although there are already several tools and documentation that might help us, it is expected to invest a significant amount of time in research since the experience within the field of data analysis is lacking.

Furthermore, this area is still not well spread around Europe as in other countries like USA being the dominant, China and other Asian countries as the Table 3 below shows. Thus assuming that European Union is lagging when compared to other successful countries in the area.

Table 3 : Country share of publications with title header "data mining" from 1989 to 2009, adapted from (Hargreaves et al. 2014)

Rank in publications (citations)	Country	Number of publications	Citations	Citations per publication
1 (1)	The US	551	4781	8.68
2 (2)	Great Britain	131	1159	8.85
3 (5)	Taiwan	104	436	4.19
4 (3)	Canada	67	547	8.16
5 (8)	China	54	187	3.46

Another challenge will be to recognize what clustering different algorithms are the most convenient for the context of Praxis Market Gap Analysis, whereas measuring and testing each different algorithm and its efficiency shall compensate the time spent rather than applying them randomly and waiting for positive and concrete results.

To sum it up, another predicted challenge will be surely associated to the different APIs available to make use of text mining techniques. Those APIs offer a wide range of different programming languages where they might be adapted, so choosing carefully the language will be also time consuming.

Finally, the main challenge of this work is the overall improvement on supply/demand matching on Praxis website, thus the main goal is to find a mechanism and combine the different and already mentioned text mining techniques in order to help us struggling the current and unsatisfactory mismatch.

2.9 Summary

As mentioned before, there are already several web applications that make use of text mining techniques in order to transform data into beautiful data driven representations, as e.g. tag clouds based on the word frequency (Wordle), clusters that share similar characteristics (Carrot2 and VocabGrabber) or a simple FoamTree (Carrot2) that aids to understand hierarchical data. All these graphical representations share a similar characteristic for the sake of bringing meaning to data, thus emphasizing the full capabilities of modern browsers.

On the contrary, these web applications are not ample and diverse enough in order to adapt them to the desired tool. The majority can only be used online whereas it is required to manually copy the text into their system, such process does not facilitate the manipulation of data within Praxis. Likewise, most of the tools, despite Carrot2, do not provide any useful API that could be potentially included and manipulated into our final solution.

In conclusion, the development of a new brand and personalized web tool for Praxis Market Gap Analysis while taking advantage of such tools, whether including or extending their functionality, is in our point of view the correct approach. The suggested approach will provide more control to the data being manipulated, better performance on the queries being made by the user and a finer adoption of the brand guide to the Praxis Market Gap Analysis web tool.

3 Research on existent tools

This chapter shows how already existent tools might benefit this work and it will also explain in detail the approach to adapt them into our final solution.

3.1 Frontend

The purpose of the frontend solution is to display all the valuable information about the Praxis Market Gap Analysis that has been already arranged by a backend solution. Its aim is to give a clear idea represented by a visual structure on what is going on between the supply and demand mismatch.

3.1.1 AngularJS

The most of the code running in Praxis is written in PHP based on the Symfony framework, however if the objective is to create a separate platform called Praxis Market Gap Analysis, we should reconsider whether we continue with the same framework or rather move to a different, newer and potentially better one. Regarding the author's experience with single page applications, it is understandable that his preference will reflect the use of AngularJS as main frontend framework. But there are also several other reasons, as e.g. if the platform acts more like an application that heavily relies on client side rendering (showing charts, tag clouds, etc.), thus it is highly advisable to maintain all the responsibility of managing state and presentation on the client side, since it will be considerable easier to maintain and more user friendly. AngularJS is quickly becoming the dominant JavaScript framework for professional web development, and there is a lot of documentation provided by Google. (Conrad 2013)

Furthermore, almost all the libraries that help to make the usage of charts and tag clouds are purely written in JavaScript.

Lastly, it is important to be aware that it is not the responsibility of a frontend solution to call and retrieve data from a database, since AngularJS will be expecting to consume such data through a RESTful web service.

3.1.2 Google graph

This library provided by Google is special since it allows the visualization in an effortless manner, it is simple to use, powerful and the most important, it is free. Its API is built in JavaScript and does not seem to be a problem to integrate within the project, hence the only required step is to embed it into the frontend application.

There is already a well written page[3] where all the required documentation is. There are several types of different charts we might want to include, however the most well-known are e.g. Bar chart, PIE chart, line chart.

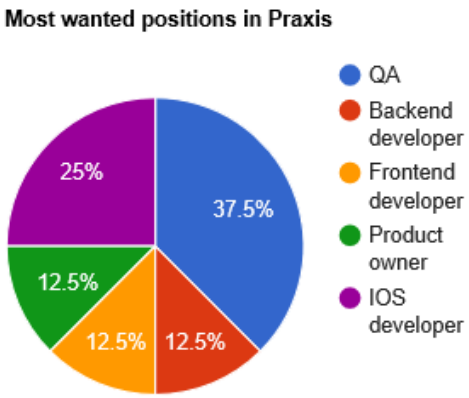

Image 6: Example of a pie chart provided by Google graph

The Image 6 illustrates how useful in terms of information this library might be. The supply and demand clusters might have a representation through a pie chart to see how big they are.

The purpose of this work is to test whether this library is fast, cross browser compatible and serves our purpose or we rather need to find different library. Lastly, it is expected to know what types of charts might be the best in terms of information to the user, since the library offers a range of different types.

[3] https://developers.google.com/chart/interactive/docs/ (visited on 17/02/2016).

3.1.3 D3.js

D3.js is a JavaScript library that transforms ordinary data into dynamic and interactive views in web browsers. D3 stands for Data-Driven-Documents and it is recognized for allowing great control over the views. (Viau 2012) In other words, D3 binds superficial data to a Document Object Model (DOM) so the user can visualize and interact with data that was meaningless before. This framework is extremely fast, it has minimal overhead and supports large documents as well as dynamic behaviours for interaction while exposing full capabilities of web standards as HTML, SVG and CSS. (D3.js 2015) It supports “modern” browsers, where Internet Explorer 8 is not included, however it is tested against Firefox, Chrome, Safari, Opera and IE9+.

The New York Times has been using this library for their rich graphs (Ashkenas et al. 2012) what somehow justifies their credit.

3.1.3.1 D3-cloud

This is a library written purely in JavaScript and it uses HTML5 canvas. The main reason to use this solution is strongly connected to the fact of being written in JavaScript as well as Google graph, thus the integration of these two libraries might be easier to handle rather than implementing them in different programming languages. Another reason is that it has a considerable API reference that is incorporated in the repository page.[4] Also the considerable number of stars and forks within the mentioned page reassembles its credibility and usage around other platforms.

Image 7: Tag cloud generated by d3-cloud library using text within our introduction

The Image 7 presents a beautiful and simple layout, it is the objective of the work to achieve such simplicity but bearing in mid the importance of the valuable information behind. This technique will illustrate how often and what keywords the users using Praxis are searching for.

[4] https://github.com/jasondavies/d3-cloud (visited on 17/02/2016).

Such technique will be used for the frontend solution, its aim is to give clear idea of which keywords are present the most during a certain period of time, thereby the user responsible for interpreting the graphical data will have a whole figure of what are trendy and not that trendy keywords used at the moment.

Finally, it should be evaluated if having word tags with different colours have better or worse performance for the user than using single colours.

3.1.4 Moment.js

Moment.js is a lightweight JavaScript date library that simplifies the process of parsing, validating, manipulating and formatting dates in JavaScript. (Moment.js 2016) It is designed to work in the majority of the main browsers, such as Chrome, IE8+, Firefox and Safari.

The reason to use the mentioned library in regard to the default Date object in JavaScript is related to the fact that the most of functions on the latter are usually insufficient and not powerful enough in terms of performance. (Czekaj 2016) It is also designed to work in Node.js environment, therefore if using a library that manipulates time it is important to maintain coherence across the backend and frontend solution.

Finally, it supports i18n (*internationalization*), l10n (localization), it is freely available under MIT license and the quality of the documentation stands out for a free tool.

3.1.5 Bootstrap

Bootstrap is undeniably the most prominent HTML, CSS and JavaScript framework for developing responsive web sites. It is used only in frontend development unlikely other frameworks and eases the process of reusing forms, buttons and other user interface components, turning web development easier and faster. (Bootstrap 2016) Essentially, it is a free collection of tools supported by a vast community of users and some of the reasons for using this framework are related to the fact that it is relatively easy to get started, it has a great grid system and an extensive list of components that in the end will save web developers' time when writing CSS/HTML code.

3.1.6 Grunt

Grunt is used as a JavaScript Task Runner for frontend web development, it allows the automation of repetitive and time consuming task as e.g. minification, compilation, running unit tests and linting. The live refresh on the browser while coding is also part of its main features. (Chu 2013)

The plugin is installed and managed via npm and even if it lacks on documentation, any JavaScript project turns to be more efficient and with better performance on its build process.

WordPress, Twitter, Adobe and other prestigious tech companies are using Grunt within their web cycle development. (Grunt 2016)

3.2 Backend

This solution will handle the connection to the MySQL database, already implemented on Praxis. It will also be responsible for processing a big amount of text data related to either proposals or searches.

The first challenge is to know what language our backend solution will be based on, whether on C#, Java or JavaScript. C# has the great advantage of having SQL-like language integrated queries suited for querying data from databases, in addition Java has the advantage of being in the same language as the API for clustering Carrot2. Finally, with JavaScript it would be way easier to manage and install all the required dependencies.

It will be tested what programming languages are the most suitable to solve the current problems within Praxis Market Gap Analysis, perhaps the language easier to adapt with the database will be chosen.

3.2.1 Node.js

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine (Node.js 2016), open-source and used worldwide by developers for server-side Web applications. The initial release was in 2009 (Node.js 2009) and its applications are cross-platform since it can run on Mac OS X, Windows and Unix servers.

Node.js uses a simplistic model of event-driven programming while leveraging the creation of highly scalable servers and allowing the development of fast web servers in JavaScript, wherefore making it accessible for a greater part of the developers, relying on the fact of being a well-known programming language. (Ornbo 2012)

The main advantage of using Node.js rather than e.g. PHP is that the former functions are designed to be non-blocking, thus optimizing the efficiency and scalability regarding input/output operations in Web applications. (Orsini 2013)

3.2.2 Npm

Npm is Node.js default package's manager and it encourages programmers to share their code among others and consequently publish it, thus simplifying the process of installing, uninstalling and updating Node.js libraries. It can be fairly described as the place where reuse is the key factor while sharing is their motto.

Node.js developers are able to find useful packages/modules for their applications and considering the community around npm it is likely that there are already packages solving the problems that the same developers are seeking to resolve. Such packages can be easily found in their official website[5].

Third-party dependencies as e.g. the connector to MySQL, the minimalist web framework for Node.js Express, Carrot2 and lodash will be defined in a file called package.json and managed by npm.

3.2.3 Lodash

Lodash is a modern JavaScript utility library that delivers modularity, performance, other extras and it is released under the MIT license. (lodash 2016) It is used worldwide by a wide range of web developers in consideration of its credit and already mentioned characteristics. It is also quite easy to install and customize it to the project's own needs.

This utility library might be extremely useful for any project since it provides many functions that work on collections and that are often needed, it is also distinguished for being focused on speed and in browser compatibility (Chrome, IE, Edge and Safari) more than the native JavaScript. (Zukowski 2013)

3.2.4 Express

Express is a Node.js web application that serves as a server framework, it is fast, simple, flexible and provides a strong set of features for web development. The most important features within Express are e.g. robust routing, focus on high performance, a super-high test coverage and HTTP helpers. (Express 2016)

Such server framework seems ideal for building a public REST API that will be consumed by the frontend solution for the Praxis Market Gap Analysis. It has also a wide variety of available plug-ins and good documentation what is highly desired when building such tools.

The installation of Express within Node's environment is straight forward because it is also managed by npm, thereby it will be effortless to create the desired web app.

3.2.5 Lingo3G versus Carrot2

Lingo3G is a text document clustering engine, it is paid and comes in three editions. The advantage of Lingo3G compared to Carro2 is that the former offers better cluster labels, fewer unclustered documents and much higher processing speed.

[5] https://www.npmjs.com/ (visited on 15/03/2016).

The Table 4 shows precisely the differences on speed processing between the free and the paid version.

Table 4: Lingo3G vs Carrot2 adapted from (Carrot2 2016)

Number of documents	**Lingo (open source)**	**STC (open source)**	**Lingo3G (commercial)**
100	0.054s	0.003s	0.005s
500	0.490s	0.015s	0.018s
1000	0.847s	0.026s	0.029s
10000	121.5s	0.218s	0.154s
10000	does not support	3.802s	1.561s

Lingo3G in general has much better results, however the outcome when there are few documents being clustered is more than sufficient for the purpose of this work. Nevertheless, the paid license for Lingo3G might be reconsidered if there is a higher demand in the number of documents being clustered.

3.2.6 Carrot2 Document Clustering Server

Carrot2 Document Clustering Server (DCS) exposes Carrot2 clustering as a REST service. It clusters documents and returns the results in either XML or JSON, however the documents provided must be directly written as an XML stream. (Carrot2 2016) The REST service will be used to cluster all the given documents based on its similar characteristics.

The main advantage of using it within Node.js is that there is already an implementation[6] for it, what turns to be really easy to make request calls to this service.

In order to expose the DCS API it is needed an external servlet container as Apache Tomcat either in Linux or Windows server machine.

3.2.7 PHP-unserialize

As the Praxis' website is purely written in PHP based on Symphony PHP framework, there was a problem within the structure of the database. The table responsible for storing the information about user's searches, called *SearchRequest*, was saving information as a PHP serialized object.

[6] https://www.npmjs.com/package/carrot2 (visited on 20/03/2016).

In order to get the user's searches keywords, the backed solution has to be responsible for fetching the data from the database and retrieving the field *sessionAllData* where the PHP serialized object is stored. However, the manipulation of such data requires the unserialization of the serialized object. That is the reason why the PHP-unserialize library is needed.

3.2.8 MySQL driver

The relational database management system of Praxis is MySQL, therefore for the sake of getting the connection between the backend solution and the database a Node.js MySQL driver[7] is required. As other libraries, the only difficulty on the installation within Node.js environment is to simply declare it the third-party dependency on Node.js package management system.

The library is responsible to create the connection to the database, query the database and finally close the connection to the database. Neither INSERT nor UPDATE commands are used on these queries because the backend solution will just be responsible for fetching and interpreting the information that already exists on the Praxis' database.

3.2.9 Pre-processing

In order to generate clusters the application will make use of Carrot2, and this API accepts both XML and JSON. Thus there is the need to test whether the backend shall pass to Carrot2 API the data in XML or JSON.

JSON is more compact and readable than XML, and in JavaScript it is possible to convert JSON to a JavaScript object without using a custom parser. The frontend will be built within a JavaScript framework, as previously mentioned AngularJS.

However, our sense leads to the use of JSON rather than XML.

3.2.10 Backend responsibilities

It is the responsibility of the backend to:

- Connect to the Praxis' database;
- Extract data from the database;
- Retrieve and process data depending on user's input, as e.g. choosing different clustering algorithms or simply selecting an interval of time in order to know what keywords were the most used;
- Generate the response in either JSON or XML.

[7] https://www.npmjs.com/package/mysql (visited on 20/03/2016).

3.3 Testing

Usually the process of testing can be difficult, time-consuming and often developers tend to resist it, yet the importance and significance of testing is vital for any project. There is already a variety of different tools that eases the process of testing, in order to test JavaScript applications. In the specific case of our frontend JavaScript framework, the Angular team powered by Google explicitly mentions that there are no excuses not to test, since they have built already many features into Angular that makes the developer's life much easier in regard to unit testing. (AngularJS 2016)

The use of such testing frameworks clearly intend to help building cleaner APIs, as well as modularized and robust JavaScript code.

3.3.1 Karma

The main purpose of Karma is to make test-driven development easier, faster and funnier. Karma can be briefly described as a tool that spawns a web server while executes source code against test code to all the connected browsers. It is essentially a test-runner and meant to be used for unit testing.

3.3.2 Jasmine

Jasmine is a JavaScript open source testing framework for behavioural driven development, it is designed to be as simple as possible and an easy-to-read syntax. It borrows the best parts from other JavaScript testing frameworks and the main reason for the existence of such framework is connected to the strength of good testing practices, easier integration with continuous build systems and easier start. (Jasmine 2016)

3.3.3 Protractor

Protractor is an end-to-end (e2e) testing framework for AngularJS applications, basically it runs the tests against the application while using a real browser and simultaneously interacting with the application as it would be a regular user. As Jasmine it was built by a team in Google for the open source community. This e2e testing framework uses Selenium Web Driver to drive the tests, however in a much simple manner, as e.g. the developer does not need to add wait or sleep statements to the test, because Protractor communicates itself with AngularJS application automatically.

This tool especially helps those applications that are big enough and are not reliable to be tested in a manual manner. Another great advantage of this framework is that it makes use of Jasmine for its testing syntax, what undoubtedly simplifies the process of writing different kinds of tests around the application.

4 Design

The importance of design in any software development's lifecycle is obvious, considering that it gives a better idea how the proposed software solution will look, how it is going to work, but foremost it avoids naive solutions whether in short or long term perspective. This chapter addresses the functional requirements, also known as it describes the behaviour of the system, followed by the application architecture, where the current architecture of the different tables for supply and demand on the Praxis website will be shown. Lastly a visual mock-up of what the platform might be will be shown in the last section.

4.1 Functional requirements

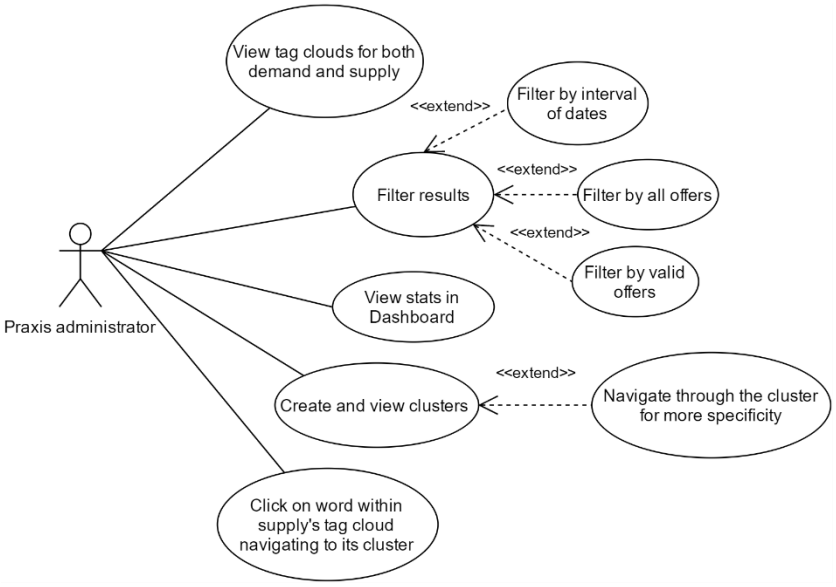

Image 8: UML use case for the praxis administrator

In order to better describe and demonstrate how the final solution will function, it was decided to make use of the use case diagram within the Unified Modelling Language, in consideration

of its information awareness and ease of use, the use case diagram is a truly simple representation on how the actors interact with the system.

The main and only actor that interacts with the system is the Praxis administrator as being the person responsible to visualize and interpret the data displayed on the web tool. The Praxis administrator has the possibility to view the current state in Praxis market using different Data-Driven Documents, either tag clouds or simple tables, on the web tool, thus allowing the administrator to take the necessary conclusions on the gap between supply and demand.

The use cases present on Image 8 are better specified in the table below that describes each use case's characteristics.

Table 5: Use cases' characteristics

Use case	**Description**
View tag cloud for both demand and supply	The administrator visualizes the most often keywords used either in user's searches or on the proposal's title, the most frequent will appear with bigger font size while the less frequent in a smaller font size.
Filter results • Filter by interval of dates • Filter by all offers • Filter by active offers	The user has the option to filter the tag clouds either by date or by type of offer, depending on which tag cloud he decides to filter. If the user decides to filter within demand's tag cloud, he may filter by an interval of dates in order to reflect when the searches were made, the default filter is for the *Last Day*, but there are three more, *Last Week*, *Last Month* and *Last Year*. The admin has always the possibility to specify the dates from start date until the end date, thus making an interval. However, if the user decides to filter the supply's tag cloud, he can only filter by all offers or by the active offers.
View stats in Dashboard	The administrator may view the different statistics across the Praxis official website using the Praxis Market Gap Analysis web tool. Those statistics are, as e.g. (View the number of searches made / View the number of keyword searches / View the number of offers / View the number of active offers / View the number of search requests with empty results / View the number of different providers / View the average number of results in searches using keywords).
Create and view clusters • Navigate through the cluster for more specificity	When the user inputs the word he wants to be clustered, the cluster will be created and all the information will be presented in detail, the user has the option to navigate through the dendrogram for more specificity (a dendrogram is the hierarchy of clusters).
Click on a word within supply's tag cloud navigating to its cluster	The administrator may click on a word within supply's tag cloud and he will be redirected to the clusters' view for the word he clicked.

These use cases demonstrate how the Praxis administrator will interact with the designed application. The main goal intended for the Praxis administrator is to detect gaps between

supply and demand, therefore two tag clouds will be provided in two different views so he can compare them and take conclusions. Moreover, the option to filter the results present in the tag clouds, provides to the administrator a better level of manipulation and interaction with the tool, what turns it to be better manageable.

The main purpose for each use case, is to give the best possible insight about what is currently going at the Praxis market, whether with beautiful tag clouds or simple numbers, it does not matter, what truly matters is that the user is able to have a clear and quick snapshot about the discrepancy in PI.

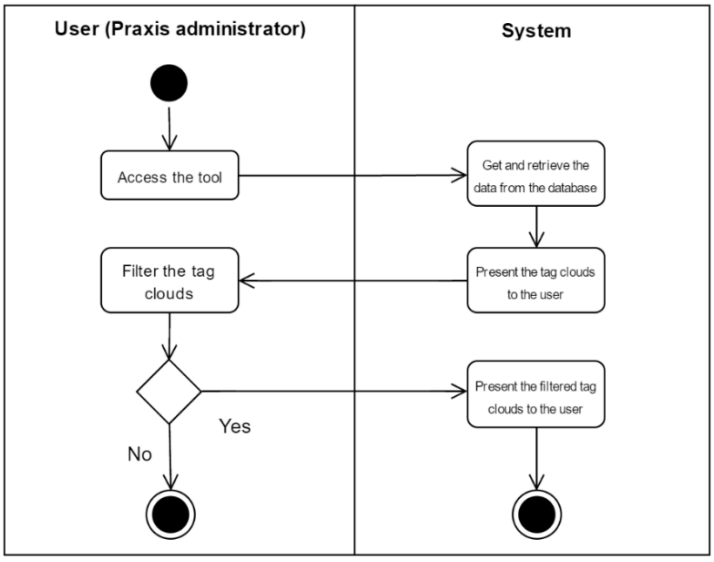

Image 9: Activity diagram representing the user's interaction with the tag clouds

As shown in the activity diagram above, the user as the Praxis administrator will firstly access the Praxis Market Gap Analysis web tool, then the system processes all the data from the database and afterwards the statistics, insight and information about Praxis market is displayed. In the specific case of the tag clouds on supply/demand, the user has the possibility to filter them either by time on the demand side or by type on the supply side. The possibility to filter the results given the interval of dates is crucial, therefore the user can check what is happening on the specific time range and check what the discrepancies that were unseen before are. Moreover, the user has also the possibility to check what are the most prominent keywords that belong to valid offers, also known as supply, that are still on the market and compare them to the keywords on the demand side.

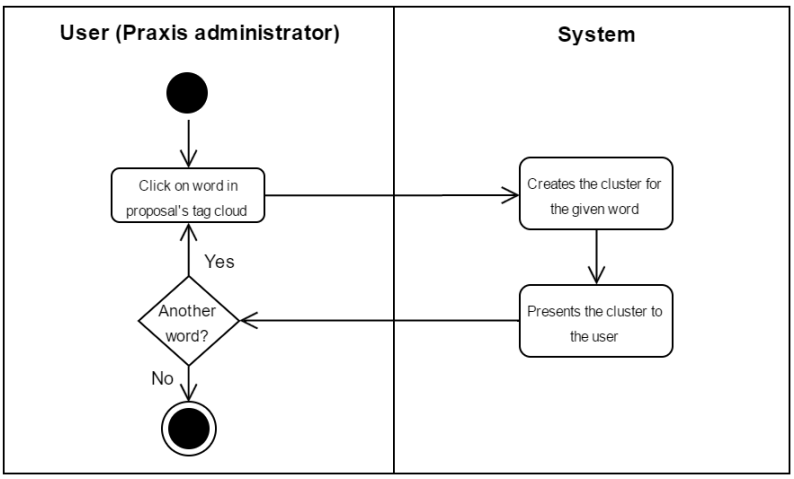

Image 10: Activity diagram representing the user's interaction for the creation of clusters

The Image 10 shows how the user can create different clusters for a given word within the tool, the only necessary step is a click on a chosen word that is present on the proposal's title tag cloud, thus being redirected to a different view where the results are shown.

4.2 Non-functional requirements

As non-function requirements, the final solution will need different servers to proper function:

- Apache Tomcat 7 for hosting Carrot2 DCS;
- Node.js' server for hosting the Praxis Market Gap Analysis backend solution;
- Access to the Praxis' production's server.

4.3 Application architecture

The diagram presented below referring to the application architecture describes how the different layers in the designed solution communicate with each other. This application architecture reflects the advantages of using JavaScript full-stack solutions as e.g. MEAN.JS, therefore increasing the chance of building fast, robust and maintainable code production. (MEAN.JS 2016) MEAN.JS stands for web applications that make use of MongoDB, Express, AngularJS and Node.js. In regard to our work, MongoDB is replaced by the already existent MySQL as the default Praxis' official website database management system. The convenience of choosing such architecture is that both server-side and client-side execution environments are written in the same language, JavaScript, therefore improving both quality and agility.

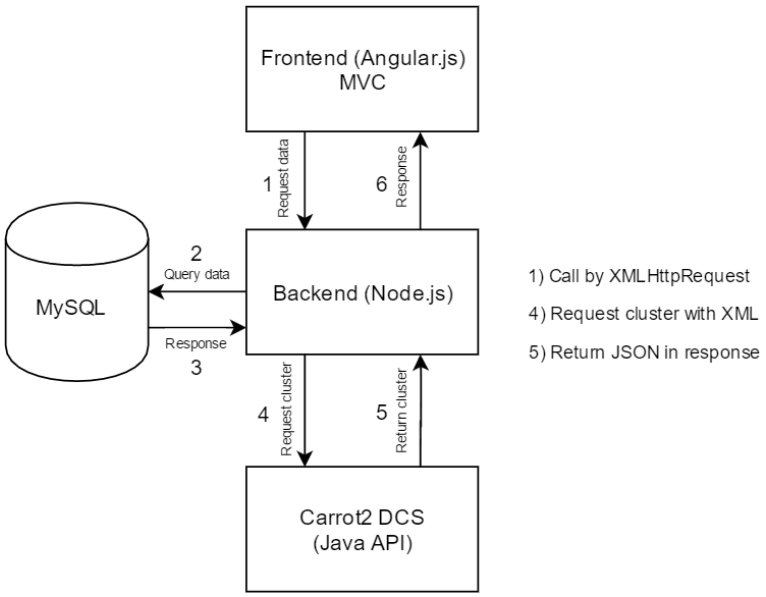

Image 11: Application architecture and the sequence of communication

In regard to the steps presented in the Image 11, the fourth and fifth step are only executed if the request from the frontend application explicitly refers to a cluster creation, otherwise it is not necessary to make a request to the Carrot2 DCS Java API. When designing an application, in our case a web app, it is important to make sure that somehow we follow certain design guidelines, also known as, design patterns. The main reason can be explained through a metaphor that says that developers and everybody in general shall not try to reinvent the wheel. It is likely to exist already a tested and tried solution to a given problem, what turns to consume less time and effort during the implementation stage if this is taken into consideration during the design stage within any software lifecycle development.

In regard to the context of this work, nowadays it is typical that most of the logic is now pushed to the client side, especially as result of XMLHttpRequest, common feature of single-page applications and embed in AngularJS, it allows partial page updates as required, therefore contributing to the presence of Model-view-controller design pattern on the majority of the web programming frameworks.

The frontend solution will be developed according to this common software architectural pattern since it is extremely used when designing web applications, therefore its consideration and application is undeniable for building the Praxis Market Gap Analysis web tool, especially when matured frontend frameworks as AngularJS have been working in order to allow the execution of MVC components on the client side.

The MVC pattern has used and broadly used in many languages and by different generations of programmers, since it helps them to write better organized code that turns to be more maintainable in the future. This architectural pattern isolates the application logic from the user's view, what supports and enhances the separation of concerns. Moreover, its essence is to support developers to decouple the final application into different and independent

components. It is curious to testify that during the last years, most of the newest JavaScript frontend frameworks have been developed, and most of them follow somehow the MVC pattern, turning programmers to face more structured and organized applications. (Google Chrome 2016)

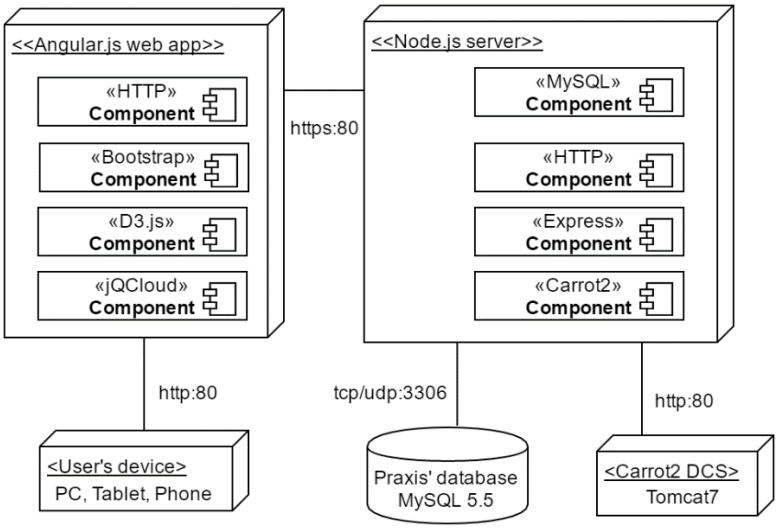

Image 12: Deployment diagram for Praxis Market Gap Analysis

As previously described MVC stands for Model-view-controller, so basically an application has three different major components. Firstly, the model is responsible for managing data and where application's data is stored. Secondly, the view is what is presented to the user and were events are present so the user can interact with the application through the view. Finally, the controller manipulates the view and updates it when model changes.

The adaptation of this design pattern within our web tool is truly important, there are many advantages, e.g. it is easier to reuse the different components afterwards, it separates view logic from business logic what strongly improves the application reusability and readability.

The following sequence diagrams demonstrate how the application works in regard to the different architectural solutions. These solutions are, firstly the frontend solution, that is responsible to manage the views to the user, the logic behind the user's interactions with UI and formulate the correct API calls. Afterwards, the backend solution is responsible to fetch data from the database and do the necessary work, in order to send it in a specific format back to the frontend solution.

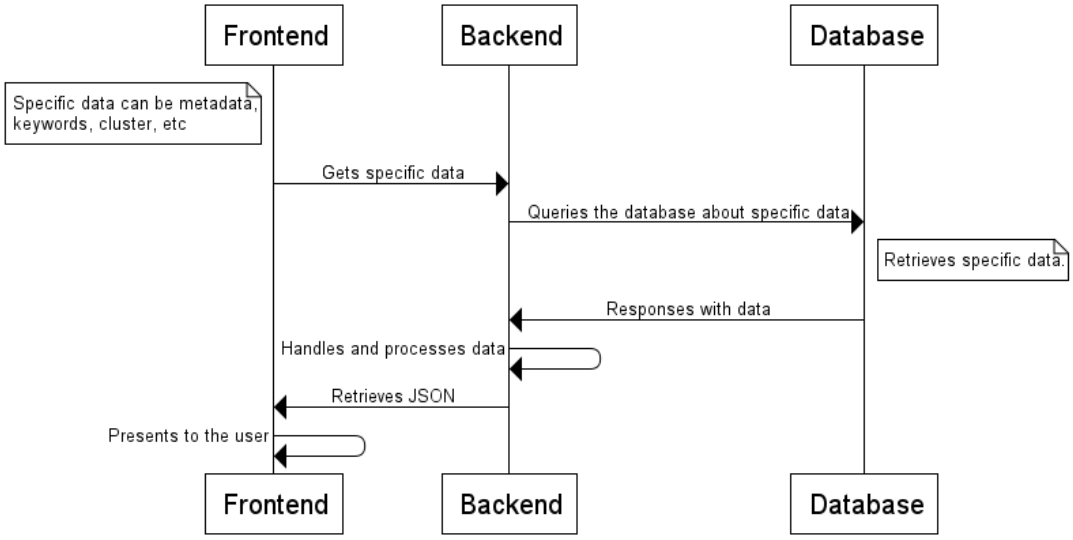

Image 13: Sequence diagram for the general retrieval of data

The image above presents a simple and general overview of how the platform will be working and what solutions are needed. The Frontend is where the user will be interacting and creating the different graphic, whilst the Backend has the responsibility to call the data from the database within Praxis and treat the retrieved data according to the specific request made from the frontend.

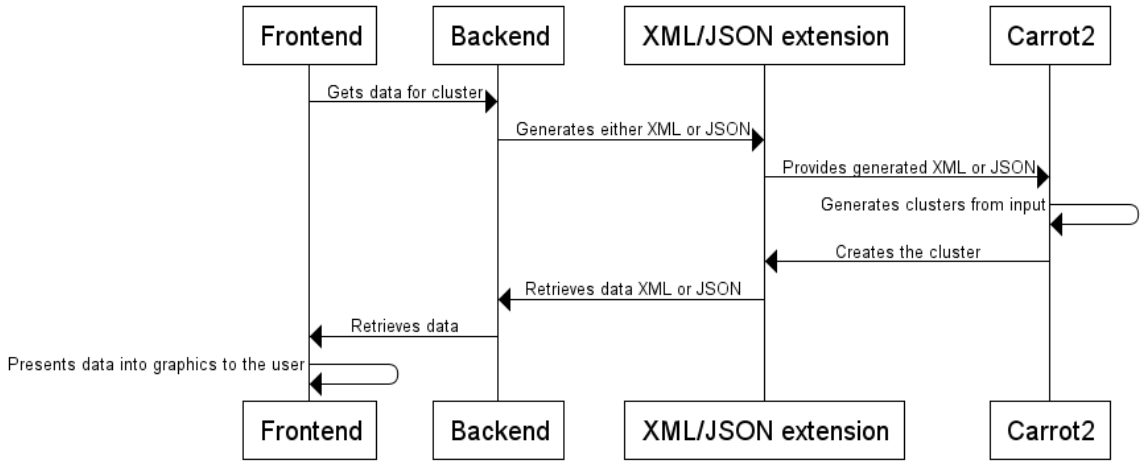

Image 14: Data generation for clusters

The Image 14 presents how the data will be generated when the user wants a specific cluster. As mentioned in the Carrot2 subchapter the solution will use the Carrot2 API, so it can generate different clusters depending on the chosen algorithm.

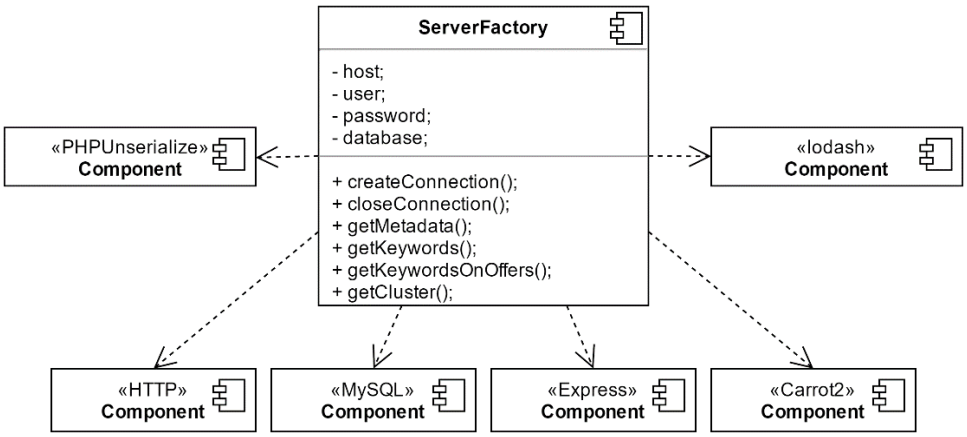

Image 15: Component diagram for ServerFactory

The Image 15 illustrates how the backend solution will look like, basically this part of the application will have its own private methods, these are the methods responsible to create and close the connection to the database. The ServerFactory is also responsible to know the information about how it can connect to the database, such as host, user, password and database's name and it is also private information, since it is only responsibility of this class to be aware about it.

Moreover, it is interesting to observe how the ServerFactory relies on other libraries, e.g. HTTP interface allows the backend solution to support many features of HTTP protocol. Express framework allows to set up middlewares, so it responds to HTTP requests. Carrot2 is responsible to transform all the JavaScript Objects into XML and send it to Carrot2 DCS with the desired parameters, so that it can interpret it. Lodash is used, as a utility library to better work with arrays with minimal effort and with great performance. MySQL is used to create/close the connection and query the database, however the only type of query used in the context of this work is the SELECT. Finally, the PHPUnserialize library will convert any PHP object into a JavaScript object, this is used because of the first structure of Praxis' database as mentioned in the Chapter 3.2.7.

4.3.1 The current Praxis database architecture

The current Praxis database has more than forty tables, however at the moment just two of them are more relevant to the project's context. The tables are the supply and demand, the former responsible for the storage of all the proposals, and the latter for the storage of all the user searches made on the website.

4.3.1.1 Supply table

This table contains all the relevant information related to the proposals that are contained on the website and it is called Proposal on the Praxis database.

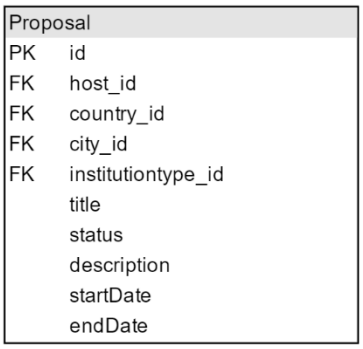

Image 16: Structure of the supply table

Regarding the structure of how the supply table is organized, there are several useful columns that will be needed to manipulate data within some certain criteria. As e.g. the column status will be only filtered by the active proposals. Then the *city_id* and *country_id* can give a whole picture for filtering them geographically in order to check which types of clusters are distributed through different countries. However, the most important field will be undoubtedly the title and the description because they pictured it out about what the offer is.

4.3.1.2 Demand table

This table stands for the information that users are searching on the website. Perhaps it is the table with most significance to the context of our project. This table represents all the input that the users are entering through the Praxis website, what keywords are they using and if results are being shown. Another important aspect is that it is possible to know on which date the user made a certain query, thereby it is possible to know what keywords are used the most given a delta time.

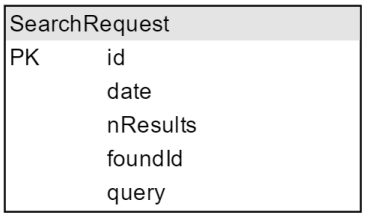

Image 17: Structure of the demand table

As shown in the Image 17 there are two fields that out-top. The *nResults* field is the variable that returns the number of result for a given query. It will be extremely important, as e.g. to know what queries are returning empty results and what queries are leading to higher number of *nResults*. Thus it is possible to track more precisely what matches are being done between the supply and demand.

On the other hand, *foundId* column is responsible for returning the ids that match with those on the proposal table. So, the number of different *foundId* in this column will be equal to the number present in *nResults*.

Lastly the query is the word set (one or more keywords) that user typed on the website. This data field within the demand's table structure is surely the most important, considering that it allows to check what was the input from the user, whereas the *nResults* and *foundId* columns are related to the output shown to the user's query.

This table is called *SearchRequest* on the praxis database and as it is possible to analyse its structure it is very simple and concise.

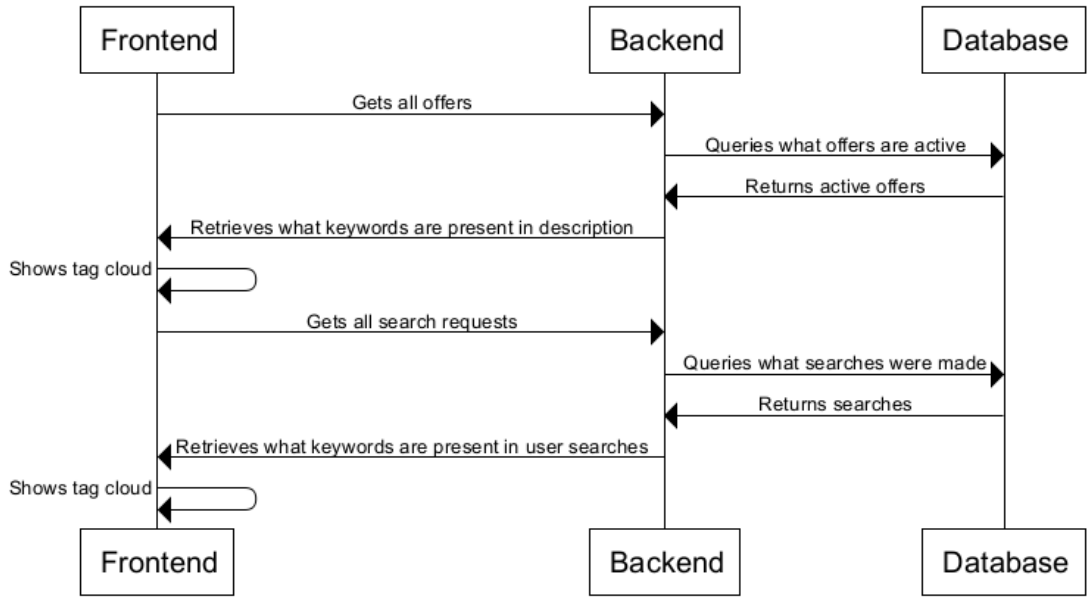

Image 18: Identifying the current gap between offers and user searches

The Image 18 presents a possible scenario where the user wants to compare the most present keywords in both offers and searches, thus identifying possible mismatches on those keywords.

4.4 Algorithms

An algorithm flowchart is used in the stage of designing a solution to a given problem. The advantage of using such graphical resources is to enhance and help the overall visualization of what is the problem about, and whether the conceived solution might have flaws or not.

4.4.1 Keywords frequency

The flowchart represented below depicts a possible and efficient solution to the problem of counting the frequency of keywords in users' searches within an array. The complexity is always linear, O(n), so it only depends on the number of items within the given array. The decision of using an HashMap is justified by knowing that this data structure is represented by a collection of keys and values, with the specific case that each key has to be unique (in our case represented

by the keywords) whereas in this example the value is the frequency of such key within the given array.

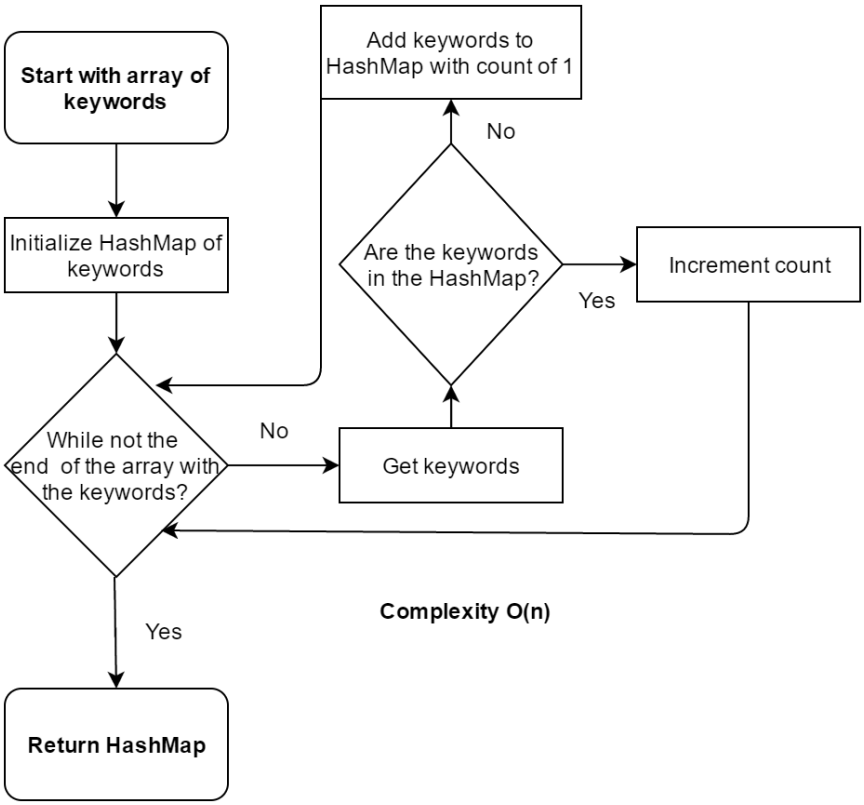

Image 19: Count frequency of keywords used in users' searches within the array

The reason to count the frequency of keywords within the array is related to the need of having structured data, so that it is possible to easily represent it using the tag cloud library. Moreover, this algorithm is specific for the case of counting the word frequency within the user's searches. In the case of offers, the algorithm shall be slightly different, in consideration to the existence of words that are irrelevant to be shown in the tag cloud and are usually presented in the proposal's title, such as prepositions, conjunctions and articles (stop words). Therefore, supposing that a title for an offer is "Internship Marketing and Business Development in Education", the words "*and*" and "*in*" are not relevant to be shown to the user on Praxis Market Gap Analysis website when accessing the tag cloud for proposals, otherwise it is obvious that such words would be the most prominent within the tag cloud, since each offer is likely to have it in the title.

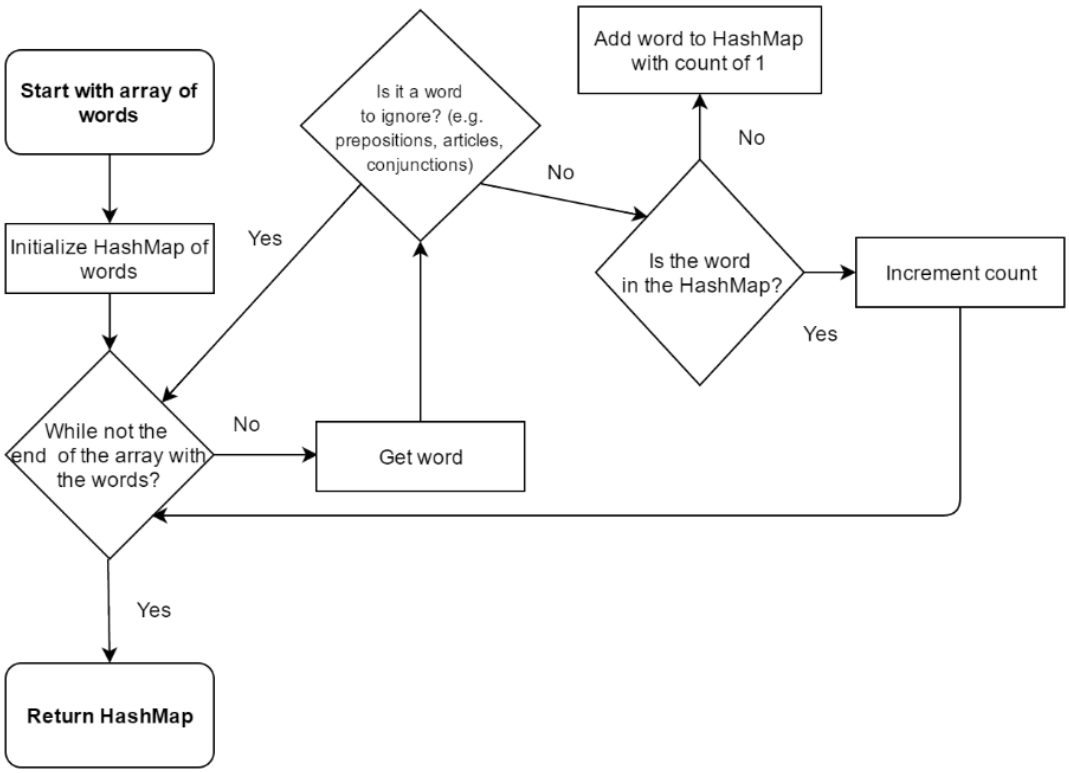

Image 20: Count frequency of keywords used in offers within the array

Thereby, in order to guarantee that all the data sent to the offer's tag cloud from the proposals' title is valid, it is necessary to make the use of a dictionary with all the words to be ignored when counting the word's frequency. In regard to the complexity, this algorithm is a bit more complex than the first, its complexity is O(n*n) since in each iteration it needs to compare the get word with the ignored words present in the dictionary.

Finally, these two flowcharts (Image 19 and Image 20) represent the general overview how both algorithms will work, in order to retrieve the required structured data for the tag clouds, afterwards it will return a collection with all the words and its frequency. The frequency will determine the size of the word within the tag cloud, so the higher the frequency number the bigger the font size of the keyword within the tag cloud.

4.5 Wireframes

In the early stage of designing and building a website, a wireframe is the ideal resource to represent the general idea of an interface, considering that it provides a visual guide, in the representation of skeleton, such that it is possible to visualize how the website will be and how its elements will be arranged. (Brown 2011)

The Image 21 presents how the Praxis Market Gap Analysis platform is. Its purpose is to be very simple and have meaningful information to the user about what is currently going on the Praxis demand versus supply data. The user will be able to generate different tag clouds based on

given dates, thus it will be possible to couple the different combinations for supply and demand. The main goal in regard to the Praxis' interface is to present to the user insightful information when he first lands to the page, therefore the first page as known as home page will have three main different areas. The first area is where the user can check the main information about the Praxis' market, such as number of searches, number of keyword's searches, number of offers and active offers, hereby having an overview of how many searches were made in comparison to the number of valid offers.

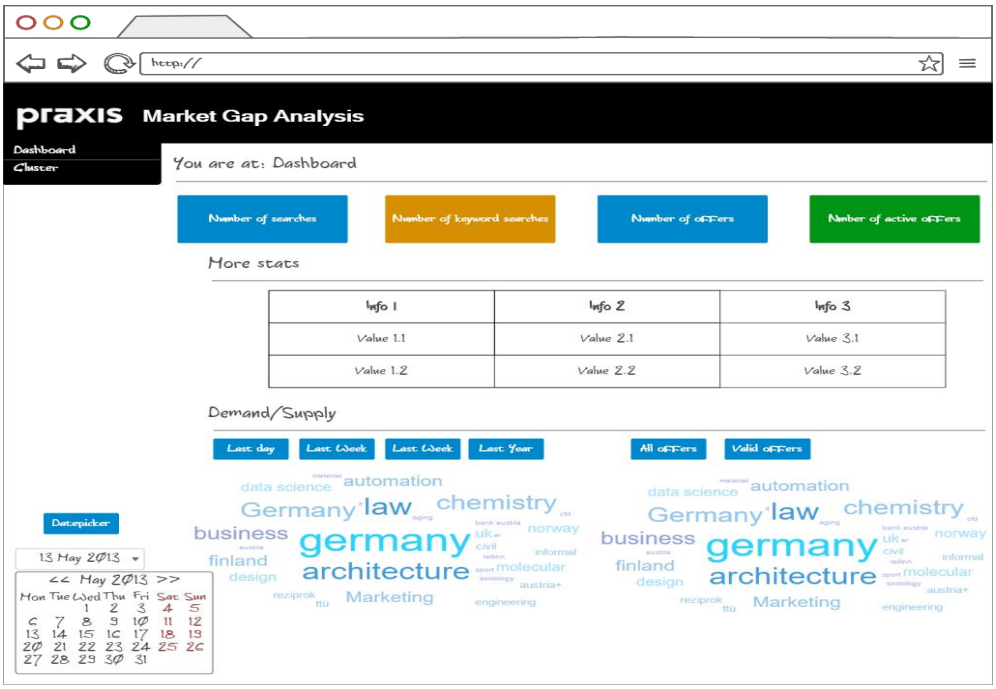

Image 21: Wireframe – Dashboard

In order to embed the most significant information in one view, there has to be the space for the second area, where all the statistics will be presented within a table. These statistics will refer to the number of different providers, number of searches returning empty results, average number of results in searches using keywords and others meaningful statistics, these statistics are better described in the section 4.1.

Finally, the third and likely to be the most important area, where all graphical insight about demand and supply is presented in the form of a tag cloud. The user is able to compare the demand and supply since both tag clouds will be displayed side by side, besides that the user may also manipulate the tag cloud, since he has the option to filter it either by date on the demand side or by the type of an offer in the supply side. These filters are the core feature of the web application and it can be justified by the power of choice provided to the user, therefore the user can check what searches were made on the current day, on the last week, on the last month or in a given interval of dates and compare them with the offers within Praxis. Another great feature within the tag clouds is the possibility to click on a keyword on the

supply's tag cloud and the different clusters and all details for that keyword will be presented in a different view, represented in the Image 22.

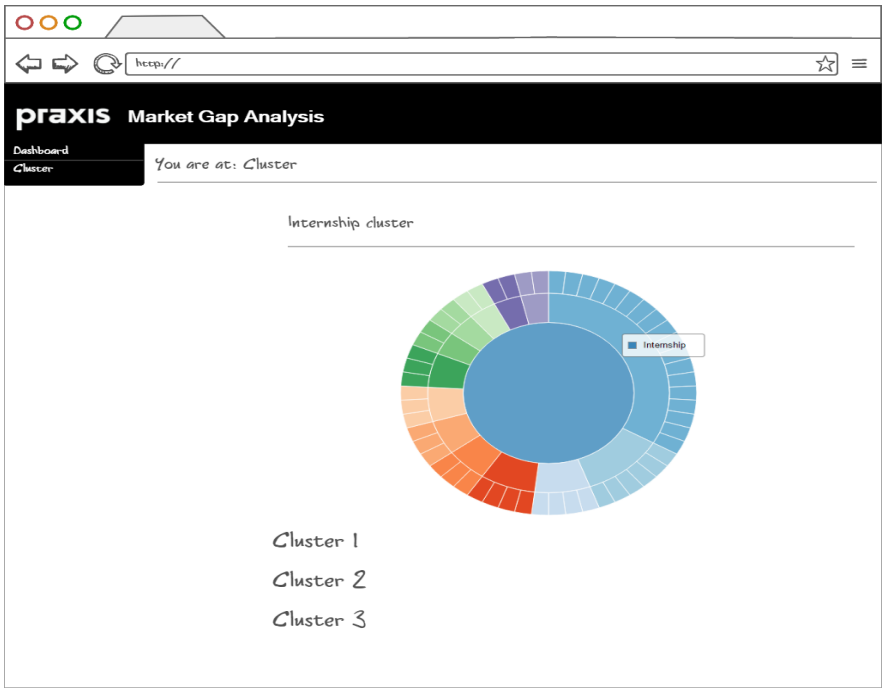

Image 22: Wireframe - Cluster

This view is responsible to display all the information about clusters, its characteristics and how they relate to each other. Firstly, it gathers all the documents from Praxis' database that contain the pressed keyword in the title, afterwards it is used a clustering algorithm that tries to find similarity between the documents and group those similar documents into the same cluster. In the end the clusters' information is represented in a sunburst chart. The decision to use this type of chart is connected to the fact that it allows the visualization of different nodes in a hierarchical manner, so the root node is the chosen keyword and the hierarchy nodes that move outwards from it, are the child nodes. The size of child nodes cannot be bigger than its parent. In conclusion the wireframes promote this tool to be a valuable resource with a pleasant interface where the user may compare the demand and supply and potentially identify mismatches within the market. It also enhances the interaction between the user and the platform's data by using Data-Driven Documents, so the user can change and manipulate what he wants to see.

5 Developing Praxis Market Gap Analysis

The whole process of developing the final solution for the Praxis Market Gap Analysis is presented in detail in this chapter. This entire chapter can be briefly described as the part where the conceived Design is implemented, forasmuch as it shows the mapping from design to the final solution, whilst always adopting the best practices within software engineering. Furthermore, it demonstrates how the application of different text mining techniques along with the most modern web technologies can enhance the quality of the information being presented.

5.1 Structure of SearchRequest table within Praxis' database

The initial structure of Praxis' database had a remarkable performance problem when saving the data within SearchRequest table that needed attention to be rectified as soon as possible. The problem was spotted when trying to fetch the specific keywords used by the users for a given search, since as previously mentioned, this table is responsible for storing the user searches' data.

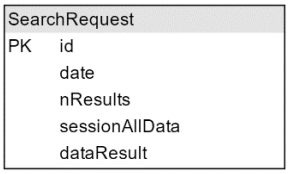

Image 23: Initial structure of SearchRequest table

5.1.1 The problem

In regard to the problem itself, it was that on each user search a big amount of data was being stored, considering that the business logic supporting the Praxis' website was storing in that specific *sessionAllData* field all the session data about the user. The data being stored was a

serialized PHP object, and this can be considered as a common and relevant mistake that shall be avoided. (Buckler 2010) Moreover, this can be justified by the fact that a serialized PHP object will not be readable in the future if there was a different and not based PHP application trying to read such data from the database, in our case e.g., a backend JavaScript application (Node.js) does not know about serialized PHP objects, hereby in order to read such data, it was required to add one more dependency to a library that could interpret it. Regarding to the MySQL disk space, the database was becoming larger and larger on each user search, therefore another great and valuable reason to change the structure of SearchRequest' table. The table, initially in February 2014, was already 1.1 gigabytes in size with only 70 966 records, this means 16.3 kilobytes of stored data per user search. Going deeper in our analysis, approximately 60 user searches were enough to store 1 megabyte of data. In a long term, the database would be unsustainable with the growing demand and it would be a heavy process to parse the data from the database, in order to generate the correct data to the demand's tag cloud, since it would be necessary to convert all the serialized data to a JavaScript object, so that the application itself could finally know what keywords were used on the user's search.

5.1.2 The proposed solution

The solution itself is very simple, in fact for the purpose of having the demand's tag cloud generated it is just necessary to store what keywords the users used on their search. Therefore, all the other data being stored in *sessionAllData* field can be discarded. This approach turns to be way simpler and cleaner since only the needed data is stored in the database. In regard to our work it also enhances the speed of processing the information, since the application will not need to parse from PHP object to a different object, the value within the field itself is already what it is required.

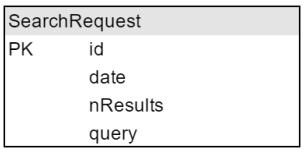

Image 24: The proposed structure for the SearchRequest's table

As it is shown in the Image 24, the table drops both *sessionAllData* and *dataResult* columns, and a new column is created, whereas its name is query and represents the specific search keywords that the user used during the time of his search.

5.1.3 Preserve the user searches in SearchRequest's table upon the change

The proposed solution is effectively the best approach to reduce the size of Praxis database. In consideration to the required space in disk for each record within the table, it is curious to mention that the size of the SearchRequest's table was determining the overall size of the Praxis' database as it can be seen in the Image 25 below.

SearchRequest	~428,333	InnoDB	utf8_unicode_ci	8.5 GiB	-
43 tables	~454,867	InnoDB	latin1_swedish_ci	8.5 GiB	0 B

Image 25: Size comparison SearchRequest's table and Praxis' database

As previously mentioned, the SearchRequest's table is mirroring the size of Praxis' database, from those 454 867 entries in the database 428 333 are from searches, whereas the number of records are not directly contributing to the problem, but rather the disk space usage on each save per user search. However, the proposed solution specified in 5.1.2 is a simple change in the structure of the table. But what would happen then to all the user searches already made on the website and stored in the database? Of course that the intention is to preserve all the necessary data rather than to simply delete all the records. It might seem tempting to keep the old data in the old format and start to store the new data in the new format, yet the old and unnecessary stored data would remain stored, what would be practically the same as not changing. Therefore, we needed a mechanism to convert all the PHP serialized objects stored in *sessionAllData* column to a known format and get the keywords searches, and finally update all those rows with the keywords whereas deleting the heavy data within the mentioned column.

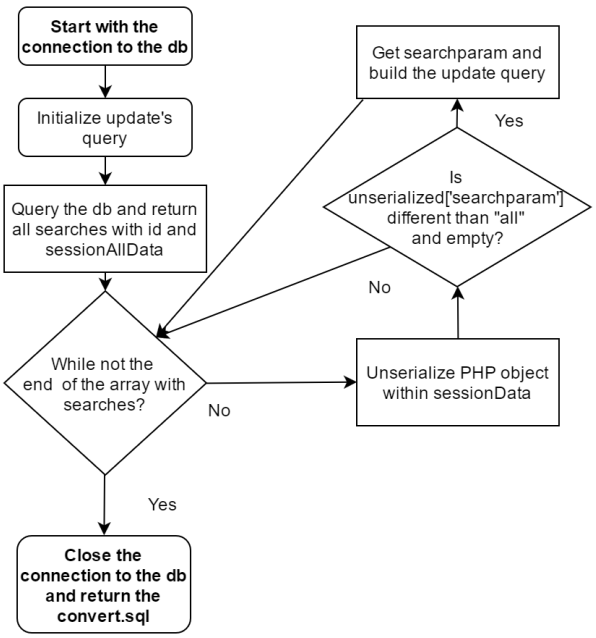

Image 26: Algorithm flowchart for converting all the legacy user searches to the new format

As represented in the image above, the core part of this algorithm is to use the php-unserialize library, in order to convert the PHP object to a JavaScript object, so we can get the search parameter used by users in their searches. When trying to use this script for the first time, in order to convert approximately all the 400 000 rows to the new format was heavy and time-consuming, we realized in that time that it would take about three days, therefore for the sake

of making it faster, we ran this algorithm in different threads, so we could take the advantage of parallel processing. The output came much faster than the initial single thread approach.

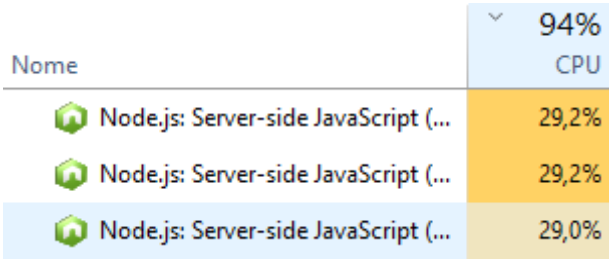

Image 27: CPU usage when three threads were running the scripting program

The CPU Intel Core i5 3210M was at 94% while running the three threads, so it was the maximum number of threads possible to run for that specific machine. Once again, these reasons suggest that the whole process of conversion was heavy and time-consuming.

The final output was an SQL file that had all the queries for updating the column query with the search parameter value identified by the search request id, as e.g. *"UPDATE SearchRequest SET query='language teacher' WHERE id=397674;".* The final SQL file had 5 megabytes with 82 657 records.

5.1.4 The conclusion

In regard to the final conclusion from this specific change to the SearchRequest's table structure. When we exported the final SQL file created by the conversion script into the database and afterwards we deleted the columns that were storing the serialized PHP objects, we faced a remarkable and impressive difference that could be clearly highlighted. The database was much smaller in size, in order to give a concrete example, from those initial 8 gigabytes of stored information, upon the application of the massive conversion update, the final size of the database was only 25 megabytes, as it is possible to see in the Image 28. It is a clear step further to the sustainability of Praxis' database as well as the Praxis Market Gap Analysis web tool. We had just reduced the size of database in 99.7%. Therefore, it will be much faster to query and fetch the information from the database, so it is possible to generate the tag cloud for the demand's side.

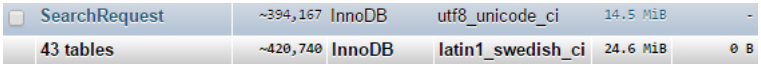

Image 28: Size comparison SearchRequest's table and Praxis' database after the change

In conclusion, this is an important change that will affect both platforms in terms of performance, whether only the required and important data is saved within Praxis' database or the process of getting the query words will be much faster and efficient.

5.2 Developing the backend solution

This chapter emphasizes how we created the backend solution in regard to the business logic requirements. In first instance, it is known that we need to gather the information from the database and occasionally send a set of documents to the Carrot2 DCS and wait to receive the organized clusters. Therefore, the backend application was developed in Node.js, and the main motivation for adopting it, it is justified by the fact of being relatively easy to install and make the use of third-party dependencies as MySQL and Carrot2. Furthermore, it is extremely easy to create and maintain web services within Node.js environment (Buecheler 2015), hereby it was clear that Node.js was the best fit for the requirements of Praxis Market Gap Analysis web tool.

5.2.1 Developing the clustering service

The clustering service is likely to be the core and the hardest feature to implement within Praxis' backend tool, the difficult part of developing this service can be explained by the requirement of having a Document Clustering Server online and supporting REST calls. The first step is to download the package's page[8] of Carrot2 DCS. Afterwards, the next step will depend whether we want to have the DCS running locally or remotely in some web server. If former, it is just to follow the guidelines present in the package's page, but basically it is to extract the package and run one .cmd file, and then the server is locally available.

However, in the case of Praxis Market Gap Analysis web tool, it is required to have it running online, so our backend solution can make REST calls to the Carrot2 DCS tool as shown in the Image 11. In order to have it online, it is required to have a Linux server distribution with Apache Tomcat 7 because of Carrot2 DCS dependency to Java. Finally, for the sake of having the Carrot2 DCS tool online is just to deploy the .WAR file into the webapps folder of Tomcat. After having the Carrot2 DCS online it is possible to start building the service itself.

The first step is to build a query depending on the word parameter, then the service will query the database for all the proposals containing the word in the title, afterwards it is just to loop over all the returned proposals and build the respective documents that can be interpreted by the Carrot2 DCS server. After having all the documents in the structure so the Carrot2 DCS server can read it, it is just to send all the documents in the REST requests to the Carrot2 DCS API, the algorithm of clustering to use and the maximum number of documents to cluster.

Finally, in regard to what was previously described, the table below presents the main input parameters when calling it via REST. The service will return a JSON with an array representing all the clusters for the given word.

[8] http://project.carrot2.org/download-dcs.html (visited on 28/04/2016).

Table 6: Input parameters for the clustering service

HTTP Method	GET
Path	/cluster
Produces	application/json
Query Parameters	Word (the word(s) to cluster)
Clustering algorithm	Lingo/STC/K-means
Maximum number of documents to cluster	Integer maxNumber

This service offers great flexibility because of its optional parameters, it is a really great advantage that the user can choose whether to use Lingo, STC or K-means when choosing the algorithm to cluster a set of documents. Moreover, it is also possible to configure the maximum number of documents to cluster.

And the image below shows what could be the output when calling this service using the word "Praxis", so all the different clusters are created and returned.


```
[
  - {
        name: "praxis",
      - children: [
          - {
                name: "Marketing",
              - children: [
                  - {
                        name: "PRAXIS Market Gap Analysis",
                        id: 170
                    },
                  - {
                        name: "Marketing plan and tools for PRAXIS",
                        id: 241
                    },
                ]
            },
        ]
    }
]
```


Image 29: Example of a response when calling clustering service given the word "Praxis"

In regard to the example above, it shows of what could be a mock response for calling the cluster service given the word "Praxis", the service queries the database for all the proposals containing "Praxis" in the title. However, what is interesting when using the clustering service, is that it tries to organize those proposals in different groups/categories, whereas the proposals in the same group always share similar characteristics, in this case it finds similarity by the proposal's title. Therefore, in the concrete example provided above, we can depict that a child group was created for the word "Praxis" called "Marketing", this means that in the case of having children, all of them must have Marketing in the title, and it is possible to prove it by the two inherited children "PRAXIS Market Gap Analysis" and "Marketing plan and tools for PRAXIS".

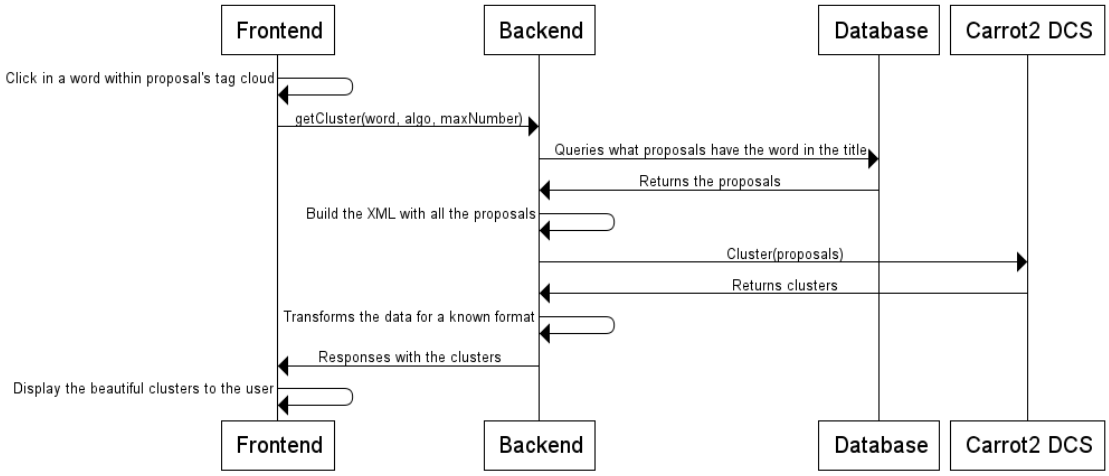

Image 30: Clustering service's sequence diagram

The Image 30 makes it easier to see the dependency over the Carrot2 DCS. As already mentioned before, the Document Clustering Server is the key for having this service working with relative ease. The integration step, in order to use Carrot2 DCS within Node.js environment was a little bit more difficult than usual, that can be justified by the lack of documentation.

5.2.2 Creating a service to expose all valid offers

Praxis's partners required a service where it would be possible to retrieve all the valid offers and its respective information. Regardless of not being connected to the main purpose of this work, this service was created at the same time we were building different services within the backend solution. The intention is to have a list of all valid offers, where each item must have the id, title, description, country and URL for each valid offer. This service will provide to the Praxis' stakeholders a simple way to access and visualize what offers are currently active on the market and what are they basically about. Thus enhancing and promoting the partnership between Praxis and its partners.


```
{
    id: 22,
    title: "Ductility of reniforced concrete structural elements",
    description: "<p>In seismic area, ductility is providing an higher final
    resistence to structural elements in reinforced concrete. The proposed project
    is aimed in carrying on testing in 1:1 scale structural elements and to compare
    their behaviour with the results of a numerical model, to be realized with in
    the Project.</p> <p>The offer is restricted to students who will apply for an
    Erasmus Placement Progr",
    country: "ITALY",
    url: "http://www.praxisnetwork.eu/proposal/id/22"
},
```


Image 31: Example of the output for validOffers service

The output as shown in Image 31 it is an array of objects, where each object corresponds to an offer. The service retrieves a JSON, so that it is easier to parse for whatever is consuming it.

Table 7: Input parameters for the validOffers service

HTTP Method	GET
Path	/validOffers
Produces	application/json
Query Parameters	Integer descriptionLength

As shown in Table 7 this service produces a JSON where all the valid offers are included. The definition of a valid offer is when its expiration date is higher or equal to the current day. In order to control how many characters are present in the offer's description, an optional parameter was added to the service, therefore when calling the service, it is possible to define the length of the description using an integer.

5.2.3 Creating a service to get all the demand's keywords

This service is responsible to fetch all the data from the SearchRequest' table, or more precisely, from the query's column. It can be briefly explained as its main purpose is to collect all the keywords used by the users on Praxis, and finally create and return a word frequency list. It can be seen an example of a response to the keywords' service in the Image 32.


```
- {
      text: "greece",
      weight: 5
  },
- {
      text: "marketing",
      weight: 3
  }
```


Image 32: Example of the output for keywords' service

As it is possible to analyze, in this example, the word "greece" had five records in the database while marketing had only three records. Therefore, the word "greece" in the tag cloud will be slightly bigger than "marketing".

Table 8: Input parameters for the keywords' service

HTTP Method	GET
Path	/keywords
Produces	application/json
Query Parameters	Date startDate Date endDate

The Table 8 presents all the input parameters for the keyword's service, as it is possible to see, the available query parameters allow to the filter the response according to the date. Therefore, if we might need to filter the keywords used by the users on a given interval of dates, these optional parameters give extra flexibility. Moreover, there is the requirement of filtering the demand's tag cloud by "Today", "Last week", "Last month" and "Last year", so the frontend

application just needs to manipulate these optional parameters, in order to have different responses.

Finally, although this service is really simple and straightforward, it adds huge value to the Praxis Market Gap Analysis web platform, since it is possible to check the most used keywords by the user upon their search on a given interval of dates.

5.2.4 Creating a service to get all the supply's keywords

This service is responsible to fetch all the data from the Proposals' table, or more precisely, from the title's column. The main goal of this service is to provide insight about what keywords are often used within the title of the offers within Praxis' market. This service has two filters, therefore it can either return the present keywords in all the offers or only return the present keywords in active offers. However, as specified in the design chapter, we need to have a stop words list. Stop words are described as words that are filtered out before or after processing of language data. (Rajaraman & Ullman 2011) Hence it is possible to remove all the prepositions, conjunctions and articles that are embed within the proposal's title, otherwise these words would be the most prominent in the response, and that is not the expected response, since it would not add any value to the supply's tag cloud.

Table 9: Input parameters for the offers' service

HTTP Method	GET
Path	/offers
Produces	application/json
Query Parameters	Boolean validOffers

In conclusion, the responsibility of this service is to gather all the titles within the filtered proposals, then it will remove all the stop words also known as the words with no relevant meaning for this specific case, and consequently will count the word frequency for each word. Finally, it will give a similar response to the keyword's service, mentioned in the previous chapter, where we have the word and its weight (representing the frequency of the word in regard to the concatenation of all the titles).

5.2.5 Allowing external connections to the Praxis' MySQL database

In the first attempt to connect our Node.js application to the production database of Praxis we got an error "*ERR_CONNECTION_REFUSED*". Therefore, we knew in first hand that external connections to the MySQL were not allowed. However, as our Node.js application is being deployed in a different machine, we needed to allow external connections on Praxis' database. Even if this technical detail is not so related to the topic of this work, it was explicitly asked to mention it, in case of future further development. Hereby, we need to login into the Praxis' production machine via SSH, give super user permissions to the user logged in, change the

directory to */etc/mysql/* and edit the *my.cnf* file, commenting the line 47 with bind-address to 127.0.0.1. Thus allowing external connections, but the last step is required, to grant the database user all the privileges to access the database in certain domain. Therefore, the query below was executed in production's MySQL.

GRANT SELECT ON praxisProd. TO root@'%.amazonws.com' IDENTIFIED BY '[password]';* (query for granting the SELECT privilege to the user root when calling from a different host, in this case, the host is ending in amazonws.com).

Afterwards, we could finally allow connections to the productions MySQL's database from the exterior, thus meaning that our application could get live data directly from production.

5.2.6 Deploying the Node.js application to the web

After finishing the development of all the services within Node.js, we needed to somehow deploy the web service to the cloud, it was necessary to deploy it to the web so that it can be reachable by other applications, as e.g. our frontend application. As already specified, the Praxis Market Gap Analysis web tool will be mainly used by one person (the Praxis administrator), so it is not required to spend that much in resources for an application that will be daily used, rather than intensively. Furthermore, we found a cloud application platform called Heroku[9] that supports the deployment of every type of application, as e.g., Node.js, Ruby, Java, PHP and Scala. Moreover, it is for free and of course that comes with some restrictions, as e.g., the application sleeps after 30 minutes of inactivity, but on the other hand it has 512 megabytes of RAM and 1 web worker, what is more than sufficient for the Praxis' requirements. The big advantage of deploying our Node.js application into Heroku is that instead of spending time on managing servers, deploying and scaling it, we could rather focus on the application itself. (Heroku 2016) Additionally, if in the future there is the need for more resources, it is easy and relatively affordable to upgrade the current plan. Finally, it is really easy to handle the deployment of an application within Heroku, we only need to create an account, install their Heroku client, and finally create the application. Afterwards, the application will be maintained in a Git repository, and the only step to push the changes to production, is as simple as pushing the changes to master (*git push heroku master)*.

Table 10: Heroku application's build log

```
-----> Build succeeded!
       ├── body-parser@1.15.1
       ├── carrot2@0.0.1
       ├── express@4.13.4
       ├── lodash@4.12.0
       ├── mysql@2.10.2
-----> Launching...
       https://praxis-backend.herokuapp.com/ deployed to Heroku
```

[9] https://www.heroku.com (visited on 20/05/2016).

As shown in Table 10 above, when the developer pushes the changes to master, the process of installing the necessary dependencies, building the application and deploying it is automatic, what saves the developer a lot of time and hassle.

Table 11: Praxis' backend configuration info

Praxis' backend web service	praxis-backend.herokuapp.com
Praxis' backend Git repository	git.heroku.com/praxis-backend.git

As shown in the Table 11 the live version of our Node.js web service API can be accessed at praxis-backend.herokuapp.com and the Git repository at git.heroku.com/praxis-backend.git.

5.3 Developing the frontend solution

This chapter explains in detail how we have developed the frontend solution using Angular.js in combination with Data-Driven Documents. Such combination was the key to better present the data to the final user, therefore enhancing the quality of information displayed in the tool. Other great reasons for this combination are the tremendous possibilities that this application offers to the user, as e.g. he can interact directly with live data coming from production, hereby getting better insight of what is currently going on within Praxis market.

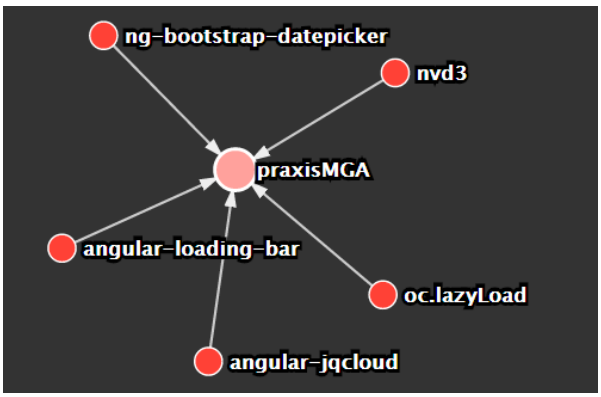

Image 33: The dependency graph of the Praxis Market Gap Analysis frontend application

The Image 33 represented above shows the dependency graph of the current frontend application to all the different modules that are being used. Of course it does not mention all the dependencies, as e.g. ui.* modules within Angular (where Bootstrap is included), since the purpose is to list all the relevant modules dependencies that we used to build the current frontend application. This graph was generated by the AngularJS graph plugin for Chrome.

5.3.1 Usage of Bootstrap

In the designing stage, we already had the feeling that we would indeed use Bootstrap. This frontend framework is free as it is open-source and provides the best templating schemas for web applications that have the need to be responsive. Although it is not a requirement for us to have it responsive, we thought it would be always an advantage. Another good reason is that the Bootstrap components are already writing in pure AngularJS by the AngularUI team. (AngularUI Team 2015) Therefore, it is really easy to setup the project to use Bootstrap within AngularJS development's environment, withal there is only the needed step to declare it as a dependency on ui.bootstrap module.[10]

5.3.2 Usage of Grunt

The usage of Grunt while developing the Praxis Market Gap Analysis web tool was surely a good asset, it saved us a lot of time when developing the tool. Moreover, the live reload when coding the application, was really a good feature in terms of efficiency, since we did not need to compile the project and check the changes by ourselves, the specific task within Grunt[11] was doing that for us. Furthermore, it was also running a task called JSHint[12] that is basically a tool that helps us to track errors and find potential problems within our JavaScript code. (JSHint 2016)

5.3.3 Usage of jQCloud

The usage of this jQuery's library was an important step further for us, since the tag clouds are likely to be the top feature to be used within our tool. This library is free and builds clean and pure HTML + CSS tag clouds. (jQCloud 2015) In addition, there was already a written directive for AngularJS [13] for the jQCloud library, therefore no time wasted to rewrite it or trying to find out a way to integrate it within Angular.js. This library is very simple to use, we can pass an array of words, where each word has to have the weight's property, and this property determines the relative importance of the word within the tag cloud.

Table 12: Word options when using jQCloud [14]

Name	**Type**	**Default**	**Description**
Text	String	Required	Text of the word
Weight	Number	Required	Size of the word

[10] https://angular-ui.github.io/bootstrap/ (visited on 20/06/2016).
[11] https://github.com/gruntjs/grunt-contrib-watch (visited on 12/07/2016).
[12] https://github.com/jshint/jshint (visited on 12/07/2016).
[13] https://github.com/mistic100/angular-jqcloud (visited on 12/07/2016).
[14] http://mistic100.github.io/jQCloud/ (visited on 12/07/2016).

The table above shows how we shall pass the information, so the directive can interpret it and finally render it.

5.3.4 Usage of nvd3

nvd3 is a project that attempts to build re-usable charts and charts components for the D3.js library. (Novus Partners 2014) This technology is used mainly to provide better and powerful insight for the financial industry, however it is also a great fit for this project. It is very simple to customize it, whereas it takes the maximum advantage of D3.js. As other libraries that were already mentioned, we considered to use nvd3 because it has also a written directive for Angular.js [15]. It is also straightforward to manipulate the data within the charts, since it accepts a JSON as the object to bind.

5.3.5 The final user interface

As per the initial requirements and wireframes, the interface for Praxis Market Gap Analysis was designed and developed in accordance with the brand guide of Praxis. The main two colours are the black and the white, these colours match with the official logo and the general schema of the official website. In regard to the platform itself, each component within this tool was thought to be very simple, but simultaneously very informative and insightful to the person using it.

5.3.5.1 The header and navigation block

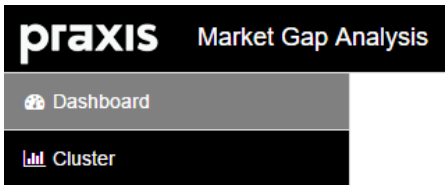

Image 34: Header and navigation block for the Praxis Market Gap Analysis web tool

The header block illustrated above was meant to be very simple and intuitive. The colours are the same as the most prominent on the official website, the white and the black. Then the official logo for the project is displayed as well as the description of what is the web application. In regard to the navigation block, there are two buttons that redirect the user to different views, whether to the Dashboard or the Cluster upon to his decision.

[15] http://krispo.github.io/angular-nvd3/ (visited on 12/07/2016).

5.3.5.2 The dashboard's view

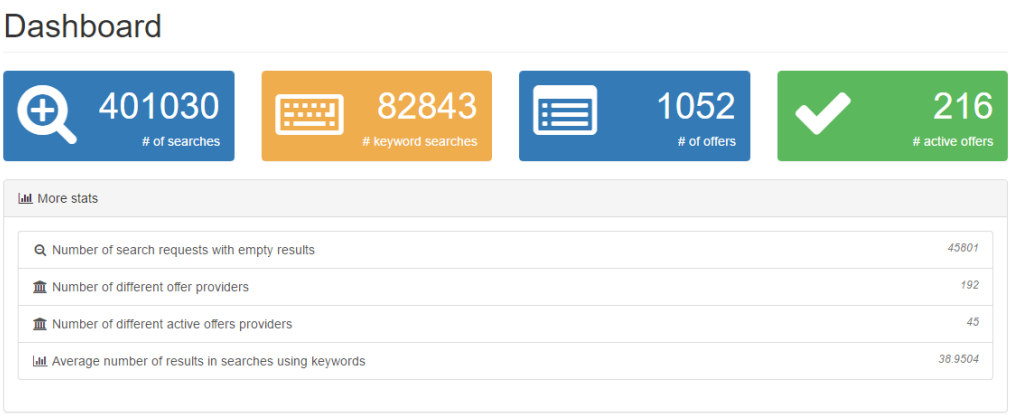

Image 35: Dashboard block in the homepage of Praxis Market Gap Analysis

The Image 35 is a representation of how the dashboard block looks like on the Praxis Market Gap Analysis' home page. We thought that it would be important to highlight in bigger blocks and in different colors the most relevant information, as the number of searches, number of keywords searches, number of offers and number of active offers. Thus, the user can already have an idea about the demand in opposition to the supply just by these variables. Downwards, there are more statistics that are also insightful and informative to the user, although the user does not need to see such information so often, therefore we decided to put it in smaller blocks in comparison to the others. Furthermore, it is important to clarify that all of this data is not result of text mining techniques, but rather simple queries to the database, that turn the information that was before only available when consulting the database more appealing to the user.

5.3.5.3 The demand/supply's view

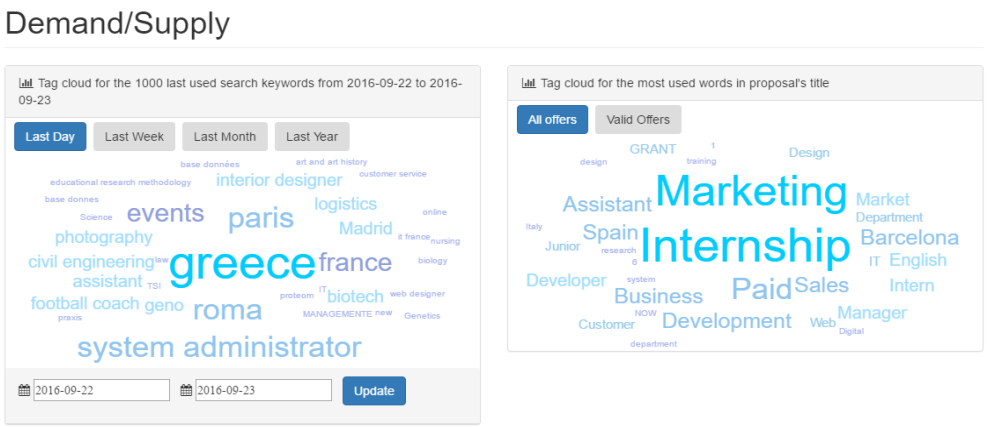

Image 36: Demand vs Supply block in the homepage of Praxis Market Gap Analysis

The block represented in Image 36 is surely the area in this tool that gives more information about the supply and the demand within the Praxis' market. The user can check, e.g. what were

the most used keywords by the word's font size for the current day and compare with the most used keywords present in the offers. If such keywords do not match, it means that there are mismatches between the supply and the demand for the today's demand. Additionally, the user has the flexibility to filter the tag clouds, he can filter the demand's tag cloud by an interval of dates, and there are other filter buttons for "Last Day", "Last Week", "Last Month" and "Last year" in order to simplify the user's interaction. On the other hand, the user can filter the supply's tag cloud by the offers within the market or by offers that are actually still valid. Thus enhancing the user's interaction with the tool, and enhancing the quality of information that he perceives. Although the tag clouds might be really simple, the value and information that it portrays to the user is really magnificent. The approach is to make the interface as simple as it can be, likewise we also developed it is as enlightening as possible. Another nice interaction feature embed with the tag clouds, is that it is also possible for the user to click on a word in the supply's tag cloud, so consequently the user is redirected for the view where the cluster for that word is displayed. In conclusion, as it can be perceived by the image above, the user has the possibility to pick the interval of dates using a date picker, what also slightly improves and facilitates the user's interaction with the user interface. In addition, as this web application is a single-page application developed in Angular.js, all the user interactions does not represent a hard refresh on the page, since it only updates the components that the user wants to see updated, what generally improves the overall user experience with the application. (Shimanovsky 2015)

5.3.5.4 The cluster's view

This is the view where the magic of using the Carrot2 API clearly spots out. As already mentioned, we decided that the best chart to graphically represent a cluster is the sunburst chart, the justification is related to the hierarchical data that comes with the process of clustering.

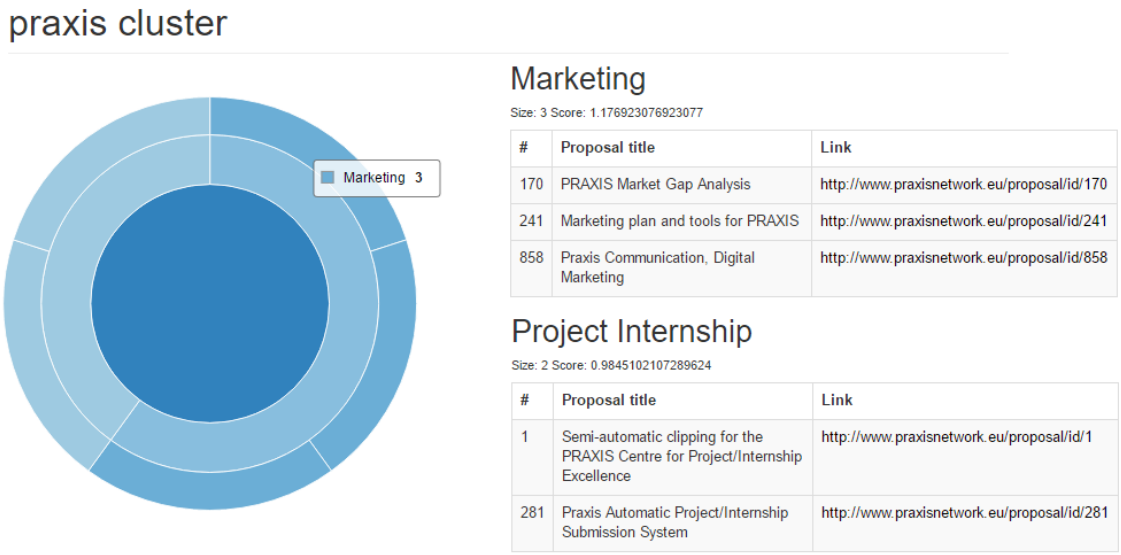

Image 37: Example of a cluster's view for the word "Praxis"

The importance of having the cluster represented as a graphical object is tremendous, because the user can have a precise idea how the cluster looks by just looking and interacting with the Data-Driven Document. In the example shown below, the user can easily perceive that the "Praxis" cluster is divided by two main categories, the "Marketing" and "Project/Internship", whereas the former category is slightly bigger than the latter. However, we decided that the graphical representation of a Cluster would be better complemented with a table, such table has more information in detail for each category. Otherwise we would overload the graphical representation with too much information, and indeed this is not our goal. Therefore, the table presents in detail all the children's information, in regard to its name, size and score, and such data is totally retrieved from the Carrot2 API.

Additionally, in each category there is a list of the offers, containing the id, title and link for each offer. Moreover, the user can have a precise idea about what offers exist for a given cluster within the Praxis market, and this adds huge value to the market analysis, because the user can clearly compare clusters by its size and type of proposals, hereby knowing precisely what type of proposals the market needs in order to fulfil the current gap.

Finally, when developing and building the clusters' view, we kept always in mind not to overload it with too much information, otherwise we would turn it into data asphyxiation. (Schumpeter 2011)

5.4 Summary

This chapter summarizes the development process and explains the decision to use a well-nigh MEAN.js application and the reason for being an advantage when putting Praxis requirements in practice.

Firstly, it was needed to gather all the application functional requirements, as specified in the 4.1 chapter, therefore before starting developing new features, we knew in first hand exactly what functionalities and behaviour to add to the application. Afterwards, we studied in detail the Praxis' database structure, we got to know the flaws within it, as shown in 5.1, hereby changing the Praxis' database structure as per our proposed solution, explained in detail in 5.1.2.

The next step was to build all the needed REST services that would be consumed by the frontend application. Hereby we analysed in detail the functional requirements, and we portrayed it into different services. Each service was responsible for a specific task, whether to analyse the frequency of keywords in the demand/supply or to process the different clustering algorithms given a set of documents. After having the web service completed, we started to build the frontend application using the framework that we have studied and we confirmed to be the painless frontend framework to aggregate all the dependencies to the different required libraries, either for adding behaviour to the tag clouds or for displaying the clusters in a pleasant way. Nonetheless, we had always in mind the wireframes represented in 4.5 when developing the frontend application, and we can state that the final result is very similar to what we have designed.

Furthermore, we really think that was a great choice to take advantage of Data-Driven Documents library, as known as D3.js, since we did not need to build by ourselves any graphical component, we just reused and adapted it accordingly to our requirements. In order to clarify, we can give a practical example, when building the cluster's view, we did not need to worry about any object transition within DOM, or when we built the tag clouds block, we did not need to give any concern about the possibility of word's intersection in the tag cloud's area. That is the main reason why we added and injected dependencies to other libraries that are intended to do that in an effective manner and with cross-browser compatibility in mind, in a much better way that we could possibly do.

In addition, we think that our application architecture is not complex, since we separated the business logic from the user interface, what is consistent to the best practices within the area of software engineering. Such separation of concerns makes easier the maintainability and possibility of further development. It is also easier to add the necessary unitary tests, as well as the integration tests, since we can isolate both applications.

Finally, it was really a step forward when we finally took advantage of the capabilities of Carrot2 API, we integrated the API in our web service, and we were amazed by the advantage of using the clustering algorithms for the specific needs of Praxis. Praxis Market Gap Analysis web tool might be simple, in terms of business logic or requirements, but it adds huge value to what was before unknown, the insight about demand and supply within Praxis.

6 Evaluation

This chapter emphasizes the evaluation that will be the part of our solution. Its purpose is to test and evaluate the adopted solution and describe what technologies will be used and the reason behind such choices. It is important to refer that the final work was a complete set of incremental deliveries that were tested and deeply analysed, thereby offering the possibility of having a complete awareness of the project's state on each phase.

6.1 Evaluating results on different clustering algorithms

As previously mentioned one of the first challenges will be based on testing how each clustering algorithm works within our context. This leads to the evaluation of the different results on the same set of data. The different algorithms that will be the part of our evaluation will be those present in Carrot2 software: K-means, STC and Lingo. The different metrics to be tested are both, the processing time for creating the cluster and also the number of clusters that do not fall into the unknown category. It is also interesting to see and test how different algorithms react to different amount of data. Furthermore, all the tests presented below were executed using real data from Praxis' database, what in our opinion, truly gives us the best idea of what algorithms to use within the Praxis Market Gap Analysis web platform.

6.1.1 The tests

In this chapter, we present the tests we have done, in order to depict and analyse what algorithm works the best for us. Moreover, we have used real data from Praxis, and we sent the set of documents to the Carrot2 DCS for clustering. All the outputs from Carrot2 DCS clustering engine can be accessed from the link presented in the footnote.[16]

[16] https://drive.google.com/file/d/0ByAFEhZd30iycmh4emF5SnJFVE0 (visited on 25/08/2016).

The first algorithm that we analysed was the Lingo algorithm, we had in consideration variables like the processing time, the number of clusters created and the size of other topics' clusters.

Table 13: Lingo testing results

Query	Number of documents	Number of clusters created	Size of other topics' cluster	Processing time in milliseconds
Sales	61	22	1	28
Business	65	24	12	21
International	77	24	13	24
Marketing	154	33	28	142
Internship	264	42	24	260

In regard to the test results for the Lingo's algorithm, it is possible to conclude that the processing time is considerable good for the number of documents that were sent. The maximum time to process was 260 milliseconds for 264 documents. The number of clusters created is also good, always more than 20 clusters, what gives us the idea that this algorithm organizes the documents into different categories that share similar characteristics in a good manner. However, the average size for the other topic's cluster is remarkable high, despite the test for "Sales", where just 1 falls into that category. All the remaining had approximately 10-20% of the remaining items in the unknown category, so the documents were not so well clustered. In conclusion, the Lingo algorithm seems somehow efficient in terms of time processing, but it collects a lot of documents within the other topics' cluster.

Table 14: STC testing results

Query	Number of documents	Number of clusters created	Size of other topics' cluster	Processing time in milliseconds
Sales	61	16	6	3
Business	65	16	8	8
International	77	16	6	6
Marketing	154	16	29	7
Internship	264	16	41	15

When first analysing the results from STC algorithm, we were astonished by the time that it took to process all the documents that we were sending, our first impression was this algorithm is extremely quick independently of the number of the documents. On the contrary, we were disappointed with the number of clusters created, because just 16 clusters were created in all the tests, it means that the clusters are considerable less specific, what is not our desired outcome. In addition, the high number of items that were falling into the unknown also known as others topic's clusters was also not so good. The "Internship" example is a good demonstration, from those 264 documents that were sent, only 16 clusters were created where 41 items fell into the unknown category. However, it is good to mention that in terms of processing time this algorithm worked the best.

Table 15: K-means testing results

Query	Number of documents	Number of clusters created	Size of other topics' cluster	Processing time in milliseconds
Sales	61	20	3	90
Business	65	22	4	137
International	77	24	2	121
Marketing	154	25	1	351
Internship	264	25	0	539

The results from K-means algorithm were not that good when compared to the STC and Lingo in terms of processing time. It is possible to see that even for a low number of documents, the processing time is quite high. However, in comparison to the number of clusters created and the number of items not falling into the unknown category, this algorithm appeared to be the very best. It is indeed impressive, that it creates a high number of clusters, what turns to add more specificity and few numbers of items fell into the other topic's cluster. So, the withdrawal from processing time really compensates the outcome.

6.1.2 Clustering algorithms' performance

The two comparison charts illustrated below are representing the data we had collected for both experimental observations, the first is the lowest number of documents sent (61 sales' cluster) and the second and last is the highest number of documents sent (264 internship's cluster). The variables concerning the chart are the number of documents, number of clusters created, processing time and the size of items that fell into the unknown category, also known as other topics.

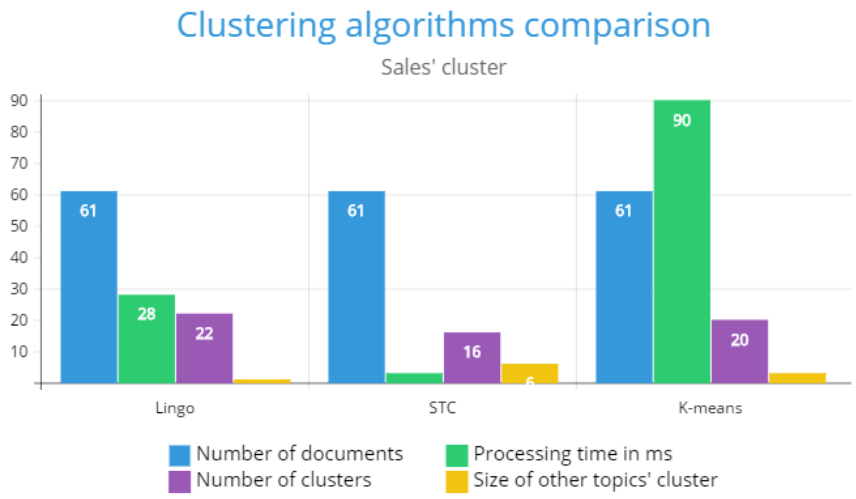

Image 38: Clustering algorithms comparison for the Sale's cluster

As it is possible to see in the Image 38 both Lingo and STC were relatively fast when comparing the time to process all the 61 documents, while K-means was a bit slower. The number of clusters created is the highest (with 22 clusters) when using Lingo, while STC has the lowest (with 16 clusters), as previously mentioned, this turns in less cluster's specificity, and it is not what we are looking for, even when the number of documents is considerably small. Therefore, it is possible to see that even with small number of documents both Lingo and K-means are the best algorithms for our needs.

However, STC is extremely fast when compared to the others, but once again, both Lingo and K-means have a reasonable processing time, in the specific case of sending bigger quantity of documents.

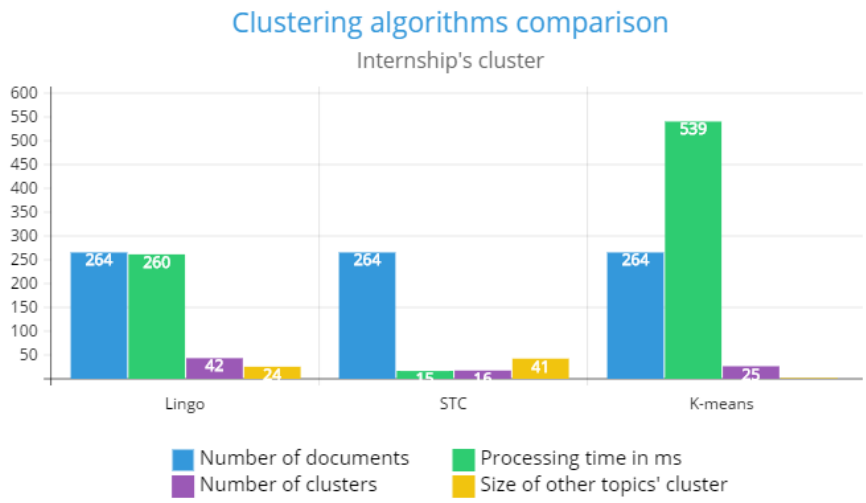

Image 39: Clustering algorithms comparison for the Internship's cluster

The Image 39 shows how the different algorithms behave with a higher set of documents. K-means is taking twice the processing time of Lingo. However, K-means has no documents within the unknown category, while Lingo has 24 documents. STC is extremely fast even with higher number of documents. In addition, Lingo creates the highest number of clusters, while STC the least. By the results, it is likely to conclude that the more documents we sent, the more processing time the K-means algorithm needs. Hereby, in the future with an ampler set of documents being sent, K-means might be considerably slow.

6.1.3 Summary

In conclusion to the clustering algorithms, we think that both Lingo and K-means are considerable good algorithms for the needs of Praxis Market Gap Analysis, since both add an extra layer of specificity within the created clusters, whereas STC is impressively fast but with less specificity. Therefore, we intended to use K-means and Lingo for the moment. The Praxis' database is not so big, since at the moment there are only 1 050 offers, and we do not expect

this size to increase that much in the future. Finally, the chosen algorithms are always the balance between the time that it takes to process and the level of specificity that it offers to the created clusters.

6.2 Evaluating number of searches that lead to empty results

In the subchapter Contextualization it was mentioned that 12.5% of searches were returning empty results, thereby it will be easy to check whether we are solving the problem or not regarding the number of searches returning empty results, since the only variable to measure is the number of searches with no results. The evaluation will be undertaken as a comparison between different intervals of time.

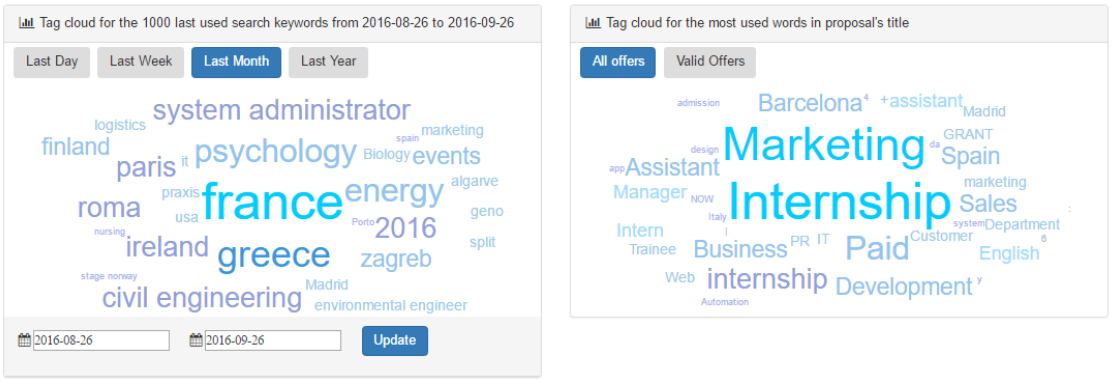

Image 40: Demand vs Supply in Praxis for the month of September 2016

As it is possible to see in the Image 40, the Praxis Market Gap Analysis web application truly spouts out the gap between the demand and supply using the tag clouds. In this example, the Praxis administrator can check that there is a high demand for “France” and “Greece” during the last month of September 2016, while in the supply market, there are not that many offers containing such keywords. Therefore, the Praxis administrator knows that he needs to contact the partners in that region, so it is possible to fulfil the current gap within the market. The represented tag clouds are a great graphical component that truly give insight and valuable information to the user. Furthermore, it is easy to interpret and to manipulate them.

Finally, it is possible to observe that the number of searches with empty results have reduced since our first analysis, from 402 151 searches, only 45 864 led to empty results (data was gathered on 26th September 2016 from Praxis database), what represents 11.4% of searches leading to zero results. From our evaluation, it is possible to state that in the last six months the number of searches leading to empty results was reduced by approximately 1%, what is a truly positive evaluation for Praxis. The Praxis Market Gap Analysis web tool spouts out the gap between demand and supply, what contributes to the increasing number of active offers that are present in the market, therefore fulfilling the gap in the demand.

6.3 Evaluating database's performance

It is a part of our evaluation to check whether we might have a limit or not on the queries we will be making to the database, since the application will basically read all the information that is present on the database, so it is expected that the bigger amounts of data we will be asking will lead to longer time to respond. The change in the SearchRequest's table structure described in detail in 5.1.2 is a great example of how we have improved the time processing to the queries made to the database, e.g. a common consultation that administrators of Praxis perform on the database is a SELECT instruction to provide the number of search requests per day. This query was taking more than 4 seconds to process before the change, however with the new SearchRequest structure it takes now 0.16 seconds. This can be justified by the current size of the database, since we have reduced from those initial 8 gigabytes to 25 megabytes.

6.4 Evaluating the added value

Although this topic is rather subjective for being part of an evaluation, we considered to be a matter of fact, so it is worth it to mention and evaluate how useful this prototype might be to the Praxis administrator. Firstly, the user can have a general idea about what is going on currently in the Praxis market, just by the dashboard's view. There the user has provided the necessary information about demand and supply, e.g., number of searches vs number of offers, therefore the user can already compare the current demand to the current supply. Obviously, such information could be easily consulted directly in the Praxis' database, but considering the time to access the database and the time to make the necessary queries, it is rather easier to have a view where all the most common consultations are provided without the hassle of doing it manually. Furthermore, our Praxis Market Gap Analysis web tool goes deeper in this analysis. It uses text mining techniques, so both tag clouds demand/supply are provided to the user, and just the analysis of the tag clouds can better shine the possible discrepancies in the Praxis' market. In addition, we think that the combination of text mining techniques with Data-Driven Documents turns to be way easier for the user to interact and get insightful information that was previously unseen. Therefore, it is likely that the difference shown in our prototype is similar to the helpdesk requests that demand more offers for a certain topic.

Moreover, with the cluster's view the user can check how the different proposals within the market related to each other and how they are characterized and organized in similar categories. It can be really productive to the Praxis administrator to have an idea how the offers are aggregated for a certain keyword, hereby having the feedback of what it needs to be changed, in order to support the demand.

In conclusion, we think that the product we have developed is extremely useful to the user, since it allows the consultation and manipulation of data, that before was only possible with manual and time-consuming work. The platform is also responsive, so the user does not need to worry in which device he shall access to get the information he seeks.

7 Conclusion

This chapter is all about the results and the following conclusions that were made during the work. It presents the relevant contributions, underlines certain limitations within the work and suggests proposals for further development.

7.1 Contributions

This project handles to give a better overview and insight about what is text mining within the vast areas in data mining and what techniques might be used in order to achieve certain goals. It also demonstrates how the different text mining algorithms can increase the chance of getting mean from a big set of unstructured text. It aims to explore the different circumstances where different text mining algorithms are more efficient. On the contrary, what algorithms are less efficient in a given context and the reason behind that. Aside from the application of the different text mining techniques, this work also relates to how we shall display the results to the user and what capabilities within modern web frameworks we should take advantage of. The most notorious example is how we aim to combine the results from mined data with Data-Driven Documents. It is important within the context of this work, not only to test what algorithms work the best, but also how the information can be better displayed to the user.

In addition, when developing this work, we faced a remarkable problem within the Praxis' database structure, more precisely in the table responsible for storing the user searches. Such table was storing big amounts of data on each user search, and we took the necessary measures to highlight the problem, suggested a solution and finally solved the issue without dropping the needed data.

Finally, we think that the Praxis Market Gap Analysis is a great tool for the Praxis administrator. It adds value to the data stored in the database, since it helps the user to understand what the current gaps in both demand and supply areas are, whereas it uses the relevant text mining techniques that we have decided to be the best fit for the purpose of this work.

7.2 Limitations

In regard to the limitations we faced when developing this work, we think as a limitation the number of searches that are not really mirroring the number of times the user has searched in Praxis, e.g. when the user has the list of offers presented in the proposals' view, and such list contains a big number of offers, a pagination section will be presented. In the case of the user clicking in a different page within the pagination section, a new search will be stored in Praxis' database. However, this behaviour is not expected and it is not a problem of Praxis Market Gap Analysis itself, we do consider it as a limitation since the web tool will show a different number of searches, in spite of showing the real number of user searches in the Praxis' search box.

Another limitation is the fact that we do not have our own servers for both Praxis Market Gap Analysis' backend solution in Node.js and Carrot2 Document Clustering Server as a Java API.

7.3 Future work suggestions

In order to amend the identified limitations presented above, we think that it would be a good asset to the work to deploy the Praxis Market Gap Analysis' backend solution and Carrot2 Document Clustering Server to a customized server that would be owned by us, therefore we would not have any restriction in terms of number of requests or time activity as we currently have using Heroku. Another consideration for further development would be to improve the actual stop words' list presented in the supply's tag cloud to contain all the words that are irrelevant for the case of analysing the current gap between demand and supply.

References

AngularJS, 2016. AngularJS: Developer Guide: Unit Testing. Available at: https://docs.angularjs.org/guide/unit-testing.

AngularUI Team, 2015. Angular directives for Bootstrap. Available at: https://angular-ui.github.io/bootstrap/.

Apache Lucene, 2011. Apache Lucene - Apache Lucene Core. Available at: https://lucene.apache.org/core/.

Apache Lucene, 2012. Lucene Implementations - Lucene-java Wiki. Available at: http://wiki.apache.org/lucene-java/LuceneImplementations [Accessed February 9, 2016].

Ashkenas, J. et al., 2012. *The Facebook Offering: How It Compares*, nytimes.com. Available at: http://www.nytimes.com/interactive/2012/05/17/business/dealbook/how-the-facebook-offering-compares.html [Accessed June 20, 2016].

Bootstrap, 2016. Bootstrap · The world's most popular mobile-first and responsive front-end framework. Available at: http://getbootstrap.com/.

Brown, D.M., 2011. *Communicating design : developing web site documentation for design and planning* 2nd ed., New Riders.

Buckler, C., 2010. Top 10 MySQL Mistakes Made By PHP Developers. Available at: https://www.sitepoint.com/mysql-mistakes-php-developers/.

Buecheler, C., 2015. Creating a Simple RESTful Web App with Node.js, Express, and MongoDB | Christopher Buecheler - Web, Writing, Cocktails and More. Available at: http://cwbuecheler.com/web/tutorials/2014/restful-web-app-node-express-mongodb/.

Carrot2, 2016a. Carrot2 | Documentation. Available at: http://download.carrot2.org/head/manual/index.html#section.dcs.

Carrot2, 2002. Carrot2 - Open Source Search Results Clustering Engine. Available at: http://project.carrot2.org/.

Carrot2, 2016b. Carrot Search: Lingo3G vs Carrot2. Available at: https://carrotsearch.com/lingo3g-comparison.

Chu, B., 2013. Grunt by Example - A Tutorial for JavaScript's Task Runner - Brian Chu. Available at: http://www.brianchu.com/blog/2013/07/11/grunt-by-example-a-tutorial-for-javascripts-task-runner/.

Conrad, A., 2013. 3 Reasons to Choose AngularJS for Your Next Project - Envato Tuts+ Code Tutorial. Available at: http://code.tutsplus.com/tutorials/3-reasons-to-choose-angularjs-for-your-next-project--net-28457.

Czekaj, P., 2016. Why should you use Moment.js and how to do it effectively. Available at: http://neoteric.eu/why-should-you-use-moment-js-and-how-to-do-it-effectively.

D3.js, 2015. D3.js - Data-Driven Documents.

David, A. & Clarence, J.M.T., 2015. *Web 3D Data Visualization of Spatio Temporal Data using Data Driven Document*,

European Commission, 2014. Mobility statistics - European Commission. Available at: http://ec.europa.eu/education/tools/statistics_en.htm.

Express, 2016. Express - Node.js web application framework. Available at: http://expressjs.com/.

Feinberg, J., 2014. Wordle - Beautiful Word Clouds. Available at: http://www.wordle.net/.

Filippov, B.S., 2014. Mapping Text and Data Mining in Academic and Research Communities in Europe. , pp.1–36.

Gary, B., 2012. Why Text Mining May Be The Next Big Thing | TIME.com. Available at: http://business.time.com/2012/03/20/why-text-mining-may-be-the-next-big-thing/.

Google Chrome, 2016. MVC Architecture - Google Chrome. Available at: https://developer.chrome.com/apps/app_frameworks.

Grunt, 2016. Who uses Grunt - Grunt: The JavaScript Task Runner. Available at: http://gruntjs.com/who-uses-grunt.

Hargreaves, I. et al., 2014. *Text and Data Mining: Report from the Expert Group*, Available at: http://dx.doi.org/10.2777/71122\nhttp://ec.europa.eu/research/innovation-union/pdf/TDM-report_from_the_expert_group-042014.pdf.

Hartigan, J.A. & Wong, M.A., 1979. Algorithm AS 136: A K-Means Clustering Algorithm. *Journal of the Royal Statistical Society, Series C*, 28(1), pp.100–108.

Heer, J. & G. Robertson, G., 2007. *Animated Transitions in Statistical Data Graphics*,

Heimerl, F. et al., 2014. *Word Cloud Explorer: Text Analytics based on Word Clouds*, Available at: http://www.vis.uni-stuttgart.de/uploads/tx_vispublications/HICSS2014_Word_Cloud_Explorer_preprint.pdf.

Heroku, 2016. About | Heroku. Available at: https://www.heroku.com/about.

IBM, 2016. IBM - What is big data? Available at: http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html.

ICEF, 2015. The state of international student mobility in 2015 - ICEF Monitor - Market intelligence for international student recruitment. Available at: http://monitor.icef.com/2015/11/the-state-of-international-student-mobility-in-2015/.

Jasmine, 2016. Background · jasmine/jasmine Wiki. Available at:

https://github.com/jasmine/jasmine/wiki/Background.

jQCloud, 2015. jQCloud. Available at: https://github.com/lucaong/jQCloud.

JSHint, 2016. JSHint. Available at: https://github.com/jshint/jshint.

Kaser, O. & Lemire, D., 2007. Tag-Cloud Drawing: Algorithms for Cloud Visualization.

Law Osiski, S., Stefanowski, J. & Weiss, D., Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition.

lodash, 2016. lodash. Available at: lodash.com.

McCandless, M., Hatcher, E. & Gospodnetić, O., 2010. *Lucene in Action*, Manning. Available at: https://books.google.com/books?id=XrJBPgAACAAJ&pgis=1.

McDonald, D. & Kelly, U., 2015. Value and benefits of text mining | Jisc. Available at: https://www.jisc.ac.uk/reports/value-and-benefits-of-text-mining.

McMonagle, A., 2013. Data storage volumes growing exponentially | Scalar | IT Solutions. Available at: https://www.scalar.ca/en/blog/data-storage-volumes-growing-exponentially/.

MEAN.JS, 2016. MEAN.JS - Full-Stack JavaScript Using MongoDB, Express, AngularJS, and Node.js. Available at: http://meanjs.org/.

Michael, B., Ogievetsky, V. & Heer, J., 2011. *D3: Data-Driven Documents*,

Moment.js, 2016. Parse, validate, manipulate, and display dates in javascript. Available at: https://github.com/moment/moment/.

Node.js, 2016. About | Node.js. Available at: https://nodejs.org/en/about.

Node.js, 2009. *node-v0.x-archive on GitHub*, Available at: https://github.com/joyent/node/tags?after=v0.0.4.

Novus Partners, 2014. NVD3. Available at: http://nvd3.org/.

Ornbo, G., 2012. *Sams teach yourself Node.js in 24 hours* First., Sams.

Orsini, L., 2013. What You Need To Know About Node.js. Available at: http://readwrite.com/2013/11/07/what-you-need-to-know-about-nodejs/.

Piskorski, J.Y. & Yangarber, R., 2013. Information Extraction: Past, Present and Future.

Rajaraman, A. & Ullman, J.D., 2011. Data Mining. In *Mining of Massive Datasets*. Cambridge: Cambridge University Press, pp. 1–17. Available at: http://ebooks.cambridge.org/ref/id/CBO9781139058452A007 [Accessed September 22, 2016].

SAS, 2015. Data Visualization: What it is and why matters | SAS. Available at:

http://www.sas.com/en_sg/insights/big-data/data-visualization.html.

Schumpeter, 2011. Too much information | The Economist. Available at: http://www.economist.com/node/18895468.

Shimanovsky, S., 2015. Multi page web applications vs. single page web applications. Available at: http://www.eikospartners.com/blog/multi-page-web-applications-vs.-single-page-web-applications.

Siegler, M., 2010. Twitter Quietly Launched A New Search Backend Weeks Ago. Available at: http://techcrunch.com/2010/10/06/new-twitter-search/.

Soukup, T. & Davidson, I., 2002. *Visual Data Mining: Techniques and Tools for Data Visualization and Mining*,

Srinivasarao, L., Kumar, R.A. & Nageswrarao, K., 2012. STC Algorithm of Clustering Search Engine with Positive and Negative Preferences. *International Journal of Engineering and Innovative Technology*, 1(5).

Viau, C., 2012. What's behind our Business Infographics Designer? D3.js of course. Available at: http://www.datameer.com/company/datameer-blog/whats-behind-our-business-infographics-designer-d3-js-of-course-2/.

Weiss, S.M. et al., 2010. *Text Mining: Predictive Methods for Analyzing Unstructured Information*, Springer Science & Business Media.

Zukowski, M., 2013. Why is lodash.each faster than native forEach? Available at: http://stackoverflow.com/questions/18881487/why-is-lodash-each-faster-than-native-foreach.