API design & implementation to connect Talentier with others

ATS

Final Degree Project
Memory

Author: Ignasi Galofré Pujol
Director: Maarten Steurs
Speciality: Information Technology
Date: November 1 2018
Gratitude

I'm thanking my parents Joan Galofré and Anna María Pujol to bring me the opportunity to study the degree I wanted and in the place where I want as well, because I could studied the same in Mallorca but they supported me to move out home.

I would also like to thank Talentier for giving me my first job experience in the technological development sector. Especially Maarten Steurs who has taught me many aspects that I will always keep in mind in the future both in software development and in team management.
1. Introduction
  1.1 Introduction
  1.2 Context
  1.3 Problem to resolve
  1.4 Involved actors
  1.5 Possible obstacles
  1.6 Personal motivations

2. State of the art
  2.1 State of the art
  2.2 Proposed solution

3. Development planning
  3.1 Project phases
  3.2 Work methodologies
    3.2.1 Scrum
    3.2.2 Scrum + Kanban
  3.3 Validation methods
  3.4 Scope of the project
  3.5 Technical skills

4. Temporary planning
  4.1 Temporary planning
  4.2 Initial planning
  4.3 Temporary development
  4.4 Dedicated hours

5. Used technologies
  5.1 Used technologies
    5.1.1 Drupal
    5.1.2 MySQL
    5.1.3 HTML/CSS/JavaScript/JQuery
    5.1.4 JSON
    5.1.5 Git
5.2 Software/Hardware used
   5.2.1 Drush
   5.2.2 Pantheon/Platform
   5.2.3 GitHub/ZenHub
   5.2.4 SourceTree
   5.2.5 Docker
   5.2.6 MySQL Workbench
   5.2.7 Sublime Text
   5.2.8 iTerm
   5.2.9 Slack
   5.2.10 Postman

6. Project development
   6.1 Phase I development
   6.2 Phase II development
   6.3 Phase III development
   6.4 Phase IV development
   6.5 Phase V development
   6.6 Phase VI development
   6.7 Final development phase

7. Economic management
   7.1 Economic management
   7.2 Resources identification
   7.3 Cost estimation
   7.4 Project cost
   7.5 Economic viability

8. Sustainability
   8.1 Sustainability matrix
   8.2 Environment sustainability
   8.3 Economic sustainability
   8.4 Social sustainability

9. References

10. Annexes
    10.1 A annexed: Talentier’s API documentation
    10.2 B annexed: Scrum + Kanban methodology rules
1.1 Introduction

This project corresponds to a Final Degree Project in Computer Engineering from the Polytechnic University of Catalonia made by B mode (projects carried out in companies). The company in which the project was made is called Talentier Lean Recruiting, S.L.

The author of the project made the Information Technology specialty. Although this project does not have the focus on the specialty performed, but it does apply knowledge acquired in some subjects studied in that specialty.

Talentier is a startup, founded in July 2015, by 3 founding partners and one investor (The Venture Gang). At the end of the year, they began hiring new members (including the author of this project) with the objective of growing quickly. While this project was made the team consisted in 9 persons; the developer team has gone from having a single programmer to having 3, including a new designer. A marketing team has also been formed. All the new members were part of agreements of different universities, which formed a young and motivated team.
1.2 Context

Talentier Lean Recruiting, S.L. objective was to change the recruitment paradigm in Spain, where the recruitment procedure is usually carried out through job offers publications in different environments and the posterior CVs received to find potential candidates. Talentier’s intention is to increase the externalizations process when looking for possible candidates through recruiters, as in the north of Europe, where there is a high percentage of these practices.

In Spain there is usually distrust or little communication between companies and recruiters. The idea of Talentier is, through a tool, to make the process more efficient and less expensive for each part, allowing to centralize and manage jobs and also facilitating good communication between companies and recruiters.

Talentier is a web service based on a marketplace, which connects companies that need to hire people through specialized recruitment consultants (or recruiters) and also an efficient collaboration tool that helps both parts to cover positions in less time and less effort.

The platform works following this schema:

1.- The company who looks for a worker post the job or imports it through some ATS, proposes a fee and choose the recruiters of the platform who wants to provide candidates for the position.

2.- The recruiters accept or reject the offer and then starts to search candidates and propose it to the companies through the platform.

3.- The client (companies) pay the recruiter if they finally contract his candidate.

Before describing the project it is important to explain that Talentier has been associated as a partner with the company Talent Clue, an ATS (Applicant Tracking System). This is a recruitment software that allows companies to automate the selection process, publishing vacancies directly on social networks, job portals and on the website, and receive all candidates in one place.

Talentier is integrated into Talent Clue like this: Once the offers have been created and published (in the Talent Clue side), it offers the option of outsourcing the search for candidates with Talentier; However, instead of doing everything from the Talent Clue itself, it redirects to the Talentier platform in which the recruiters are selected and the rates are proposed, finally return into Talent Clue (where the CVs sent by the recruiters will be managed).
On one hand, the project consisted in the design and implementation of an API for the integration of Talentier with any ATS. On the other hand, the project was focused into replace the current connection with Talent Clue to use the new API and redesign the current integration (which had several usability problems) to make it more visual and intuitive.

Talentier intentions were to grow faster. With this intention he made an important investment in development and marketing to find new customers / users.

Through the Marketing department, Talientier opened a blog at the beginning of the year to achieve a good SEO position that would allow him to receive more visits and achieve leads. These leads are followed by Hubspot and Google Analytics.

Regarding the technological part (to which this project was destined), through the creation of the API, Talentier intends to integrate with new ATS, since a large part of the users of the platform come from the integration with Talent Clue.
1.3 Problem to resolve

Previously there were two problems with the platform installed in Talent Clue.

In the first place is the use of the current pop-up, even being functional, made the Talent Clue platform not to much usable, since it seemed a totally different tool within a small compressed pop-up, so it had to be improved.

On the other hand, it was detected that the incorporation of Talentier to other platforms was not scalable, since for each ATS it was necessary to apply adaptations. So, the intention was to make the tool as easy and quick to adapt with any other ATS. For that, it was thought to unify all the necessary utilities in an API.
1.4 Involved actors

The involved actors in this project have been:

● **Project's director:**

Maarten Steurs, Talentier's CTO. He was previously the only developer of the company and is the creator of the current connection with Talent Clue. Its mission was to explain to the author of the project the current architecture of the existing connection and give support to establish the architecture of the API to be carried out.

● **The developer:**

Ignasi Galofré, the author of the project. Degree student in Computer Engineering. In charge of designing and developing the API, following the guidelines and times set at the beginning of the project.

● **Talentier:**

The platform objective is to allows the integration of new ATS and reach new customers, who thanks to the creation of the API have the possibility to open new job offers from the established connections with other platforms.

● **Talent Clue:**

Has been the first ATS to integrate de Talentier's API and is with the platform through have the tests been done.

● **The users:**

The potential users of this integration are new companies who works with ATS.
1.5 Possible obstacles

- **Project bounded in time:**

  The project was limited in time, since it had a maximum delivery date, so a good planning was necessary. The project suffered changes, but there was enough time to finalize it.

- **Previous unknowledge of the technology:**

  Talentier works with Drupal, a framework with a high learning curve. Apart from Drupal, also worked with other new technologies which has to be learned as well.

- **Bad initial design:**

  The possibility of doing a bad design at the beginning would mean lose developing time.

- **Contact with others ATS:**

  Once the project was finished, we wanted to incorporate more ATS to the web service, since this project is not only done to improve the interaction with Talent Clue. In this sense, a possible obstacle was that of not being able to prove the result on another platform. Now, once the project is finished, the platform has not yet been tested with another ATS, so 100% can not be guaranteed to be functional on any platform, but should be.

- **Limitations in human resources:**

  The fact that the developer was a single person this supposes more workload than being a team. So, there was a risk of not having time to finish all the tasks planned in the beginning.

- **Talent Clue’s dependency:**

  To test the API with Talent Clue, the coordination of the two parties was necessary. In fact, an example of this problem was manifested by having to postpone the final test a couple of days.
1.6 Personal motivation

Despite having done the specialty of Information Technology, what I really love is the development of software and that is why all the optional subjects that I have chosen are from the Information Systems specialty.

Before starting university studies it was clear to me that my goal was to train for a career once I had finished a project on my own. And that's why when I saw the opportunity to do the Final Degree Work in a startup I didn't doubt to post my Curriculum in Talentier. Now I'm living in Thailand where I'm starting my first own project.

Once I was proposed several projects I chose this in particular since it was the one that more complexity in terms of previous study, design and development had. This made me think that I could put into practice a good part of the knowledge learned in the university.
2.1 State of the art

Once seen the main problem of Talentier was the scalability (which is one of the main disadvantages of any startup), it was decided to devise an API that could be integrated into different recruitment platforms.

The purpose of an API is to allow communication between software components by supporting the functions of other services, which allows the abstraction of the code. With the adaptation of the present project, it is intended that the platform can be integrated into different ATS to outsource the search for candidates with registered recruiters in Talentier.

For this, we must first study the architecture of the existing connection with Talent Clue. Both Talentier and Talent Clue work with Drupal, a factor that makes integration with easier. However, the goal is to incorporate an API to the current platform that not only works between platforms based on Drupal, but can be integrated with any existing web technology.

Due to working with Drupal, there are modules that can be used and modified as you like, so there were many ways to perform APIs with Drupal. Currently there are 4 modules: Services, RestWS, Endpoint and RESTful.

Currently there are many APIs. An example is Google services (Google Maps, Google +, etc.). Most web services currently offer their own API to be incorporated into other services.
2.2 Proposed solution

The solution to the problem of scalability was to change the integration with Talent Clue by building a structure that could be used in any other ATS platform. The old integration made use of a module that can only be integrated into other Drupal platforms (Talent Clue also uses Drupal); that is why we searched a way to do the abstract service and thus solve the problem of scalability, making it possible to incorporate the Talentier platform to any ATS. The structure that allowed to carry out this process was an API.

The solution to the pop-up incorporated in Talent Clue was the complete redesign of this through the creation of a Drupal template so that the views within the pop-up were different from those of the Talentier website. Also, certain changes were applied: the size was increased (since it was previously relatively small) and usability errors were corrected in buttons.
3.1 Project phases

The project lasted approximately 5 months, from the end of January to the end of June 2016. The process started on January 15, the date of the author of the project where started working at the Talentier company. The deadline for delivery was June 20.

The project was divided in seven phases, plus a final one not contemplated at the beginning. Those phases were:

**Phase I: Environments & Drupal introduction**

First of all the installation of the environment, both Talentier and Talent Clue, had to be done. In addition, a process of familiarization with Drupal was necessary. On the other hand, the practices in the company included a parallel work in Talentier apart from the realization of the project.

- **Phase II: GEP**

  In this phase there was a stop in the development of the project, due to the fact that among the normal work to be done in Talentier, 6 deliveries of the GEP subject had to be carried out, planned in a short period of time (5 weeks). For this reason, during the working hours in Talentier, the Drupal learning part of the first phase continued to be carried out.

- **Phase III: Design & study**

  The project was resumed the week after the end of GEP, starting with the study of the current platform and the design of the API. This phase consists in defining well all the possible communications between platforms and all the information that you want to deliver in each request.
● **Phase IV: Implementation**

Once the API was designed, it was developed, following the guidelines planned during the study. This phase was the longest, since it includes the previous time of study of new technologies and use it to achieve the proposed objectives.

● **Phase V: Substitution**

Once implemented, the previous Talent Clue communications were replaced by new requests to the API. In addition to the redesign of the pop-up that integrates Talentier in the Talent Clue platform.

**Phase VI: Testing & debugging**

This phase consisted in verifying that all communications and processes carried out between platforms continued to work correctly and also in correcting any failure found.

● **Phase VII: Delivery**

This phase consists of two different tasks: the writing of the report and the preparation of the final exhibition.

● **Phase VII: English Delivery**

To achieve the third language recognition, this project has been translated and presented in English.
3.2 Work methodologies

When the project developer (the student) started to work in Talentier, the agile SCRUM methodology (consisting of planning shorter and more concrete tasks, in contrast to the longer and more complex phases) was used. These small tasks are distributed in sprints of one or two weeks, depending on the amount of work that each one has and the specific objective that is pursued in each case.

3.2.1 Scrum

Several meetings where planned weekly:

- **Sprint Focus:**
  
  Where the team decides in what it’s going to work in the next sprint.

- **Ticket Definition:**
  
  Meeting where the tasks where defined and estimated.

- **Sprint Planning:**
  
  Assign the tasks that were going to be done.

- **Sprint Review:**
  
  Observation of the work completed up to the moment that allowed small changes in the tickets in the cases where the investment of time supposed was not correct.

- **Sprint Retrospective:**
  
  Done at the end of the sprint to share opinions about the work done and try to fix some problems that occurred during the sprint or improve some points.

- **Demo:**
  
  Every few sprints, the job was shown to the non technical team.

Even following a rigorous methodology, in mid-April the development team detected that the deadlines had not been achieved. Three main problems were found: poor definition of tasks, tasks with too great a scope and difficulties in
establishing deadlines. In fact, until then, the sprints had been lengthened to the point where the SCRUM methodology had been discontinued.

As a result of this problem, the team decided to create its own methodology, which is currently being used. It must be said that the group has noticed a difference in productivity once this new methodology has been established.
3.2.2 SCRUM + Kanban

At the beginning of May the team started to use this new methodology, based on the next points:

- There is no deadline to update the data for the production section; every day at the first hour the news of the previous day are uploaded.

- Tickets can not reach 8 story points (8 is considered more than one day of development). For this reason, large tasks have to be divided into subtickets.

- All developers are responsible for developing, reviewing the code of others and testing the tickets of others.

- Any error that is found on a ticket must be communicated to the team quickly, whether it is a definition error or a problem in the development. In this way it is intended to resolve failures as soon as possible and avoid those that may occur later.

Added to these points, those meetings were added:

- **Daily Stand-up**: Every day at 9.30 a meeting of a maximum of 15 minutes in which the metrics are reviewed to see if there is any task that blocks the process and take action quickly.

- **Refinement**: Each Monday the backlog tickets are defined or redefined and prioritized. Thus, each time a task is completed, the group knows which one is next.

- **Retrospective**: Every two weeks, the whole team is called to make verifications regarding the methodology and the work done. In this way, the need to apply changes is easily detected.
3.3 Validation methods

With the SCRUM methodology, at the end of each sprint, all the tasks were checked and, if one had not been completed - or started - the reason was studied and the priorities were analyzed, to assign this task to the next sprint, or to modify the deadline of this sprint.

In addition to the verifications, at the end of the sprint meetings were held with the director every one or two weeks to discuss the status of the project and the progress observed, and also to verify that the procedure was being carried out over the scheduled time.

With the SCRUM + Kanban a graphic it's done every morning before the Daily Stand-up, to validate the work done.

Graph with statistics of SCRUM + Kanban

- **Closed**: Means the story points of the tickets already into production.
- **Release - TC**: Tickets ready to be deployed into Talent Clue production.
  - **Release**: Tickets ready to be deployed into Talentier production.
- **Test**: Shows the tickets pending to be tested.
- **Review**: Shows the tickets ready to be reviewed.
- **Integration**: Means the tickets pending of other tickets that have to be done.
- **Doing**: Tickets working on.
- **Ready**: Backlog tickets ready to be done.
With this kind of graphics the team can validate:

Work to do

Work in progress
Time between the beginning and the end of the work
3.4 Scope of the project

The scope of the project is divided in two:

To begin, a study of the initial state of the platform was made, an analysis of it and a definition of the architecture from the beginning, taking into account the level of abstraction required and the scalability of the system. Then the design and implementation of the API was done so that it could be used in different ATS.

On the other hand, the incorporation of the use of the API in Talent Clue was proposed, together with the redesign of the integration with a pop-up in the Talent Clue platform to correct the existing usability problems.
3.5 Technical skills

CTI1.1:

Currently the company is growing and is considering the change of the hosting service, which will be one of the aspects to be taken into account before the implementation of the project.

CTI2.2:

Being a small organization, among all the components of the development team the working environment of the entire company is maintained. In addition, during the process an environment has been created with Docker for future members of the development team.

CTI2.3:

During the development of the project, safety issues are considered at all times.

CTI3.1:

The project consists of creating a web-service, which reflects the integration of this competence in depth.

CTI4:

In Talentier the agile SCRUM methodology is used, which facilitates and speeds up the work of the development team. On the other hand, the function of this team is the realization of a product previously defined in a process divided into brief tasks.
4.1 Temporary planning

The temporal planning suffered small changes with regard to what was proposed at the beginning of the project. In this section we will see the deviations occurred, the causes of these and the times used in each phase.

4.2 Initial planning

At the beginning of the project, the planning shown in figure 1 and figure 2 was made. In the first figure you can see the Gantt chart of the first two phases (gray time bar indicates the additional developer work that is done in the Talentier company).

![Gantt chart image](image)

**Figure 1. Gantt phase I y phase II**

On the second one the gray bar still appearing, and also the other phases estimations.
4.3 Temporary development

Then, in Figure 3 you can see the real times of the development of the project. The first two phases followed the estimated planning, but it was from phase 3 when the planning suffered deviations.

Figure 3. Real Gantt of the project planning

The third phase required three more days than planned, which was transmitted to the rest of the planning.
In the fourth phase, there was a need to study a Drupal module to help the realization of the API. The implementation ended a week later than planned, which meant using an action plan (which consisted of investing more hours in the tasks that were missing).

The fifth phase was carried out with the expected time, but accumulating a week of delay, which meant that the sixth phase was carried out in less time than expected. However, during the initial planning it was not taken into account that with the scrum-canvas work methodology, the testing and debugging of the tickets were carried out before being considered finished. That allowed that the sixth phase could be reduced in time.

Finally, we had to add a phase that had not been raised at the beginning, which occupied all the time remaining until the delivery date. This phase is what consisted in documenting the project and preparing its presentation.

### 4.4 Dedicated hours

<table>
<thead>
<tr>
<th>Phase I: Environments</th>
<th>Start</th>
<th>End</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Talentier’s Environment</td>
<td>15th of January</td>
<td>19th of January</td>
<td>21 hours</td>
</tr>
<tr>
<td>Talent Clue’s Environment</td>
<td>19th of January</td>
<td>21st of January</td>
<td>15 hours</td>
</tr>
<tr>
<td>Drupal</td>
<td>22nd of January</td>
<td>18th of February</td>
<td>120 hours</td>
</tr>
</tbody>
</table>

Hours per task Phase I

<table>
<thead>
<tr>
<th>Phase II: GEP</th>
<th>Start</th>
<th>End</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st Delivery</td>
<td>19th of February</td>
<td>29th of February</td>
<td>20 hours</td>
</tr>
<tr>
<td>2nd Delivery</td>
<td>1st of March</td>
<td>7th of March</td>
<td>15 hours</td>
</tr>
<tr>
<td>3rd Delivery</td>
<td>8th of March</td>
<td>14th of March</td>
<td>15 hours</td>
</tr>
<tr>
<td>4th Delivery</td>
<td>15th of March</td>
<td>17th of March</td>
<td>5 hours</td>
</tr>
<tr>
<td>5th Delivery</td>
<td>18th of March</td>
<td>24th of March</td>
<td>15 hours</td>
</tr>
</tbody>
</table>
### 6th Delivery

<table>
<thead>
<tr>
<th>Start</th>
<th>End</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>25th of March</td>
<td>31st of March</td>
<td>25</td>
</tr>
</tbody>
</table>

### Hours per task Phase II

### Phase III: Study & Design

<table>
<thead>
<tr>
<th>Start</th>
<th>End</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>4th of April</td>
<td>8th of April</td>
<td>15</td>
</tr>
<tr>
<td>Design</td>
<td>8th of April</td>
<td>15th of April</td>
</tr>
</tbody>
</table>

### Hours per task Phase III

### Phase IV: Implementation

<table>
<thead>
<tr>
<th>Start</th>
<th>End</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>18th of April</td>
<td>21st of April</td>
<td>25</td>
</tr>
<tr>
<td>Module study</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Hours per task Phase IV

### Phase V: Substitution

<table>
<thead>
<tr>
<th>Start</th>
<th>End</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>25th of May</td>
<td>31st of May</td>
<td>8</td>
</tr>
<tr>
<td>API Substitution</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Hours per task Phase V

### Phase VI: Testing & Debug

<table>
<thead>
<tr>
<th>Start</th>
<th>End</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>6th of June</td>
<td>8th of June</td>
<td>5</td>
</tr>
<tr>
<td>Testing</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Hours per task Phase VI

### Final Phase: Delivery

<table>
<thead>
<tr>
<th>Start</th>
<th>End</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>11th of June</td>
<td>20th of June</td>
<td>50</td>
</tr>
<tr>
<td>Memory</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Hours per task Final Phase

<table>
<thead>
<tr>
<th>Start</th>
<th>End</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>21st of June</td>
<td>27th of June</td>
<td>15</td>
</tr>
<tr>
<td>Presentation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>Hours</td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>Phase I</td>
<td>156 hours</td>
<td></td>
</tr>
<tr>
<td>Phase II</td>
<td>95 hours</td>
<td></td>
</tr>
<tr>
<td>Phase III</td>
<td>50 hours</td>
<td></td>
</tr>
<tr>
<td>Phase IV</td>
<td>150 hours</td>
<td></td>
</tr>
<tr>
<td>Phase V</td>
<td>28 hours</td>
<td></td>
</tr>
<tr>
<td>Phase VI</td>
<td>25 hours</td>
<td></td>
</tr>
<tr>
<td>Final Phase</td>
<td>65 hours</td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>569 hours</strong></td>
<td></td>
</tr>
</tbody>
</table>
5.1 Used technologies

The technologies used during this project were:

5.1.1 Drupal 7 (PHP)

The Talentier’s website is developed in Drupal 7, a PHP web framework.

Drupal is open source; the first version was launched in 2001. This makes it a very mature tool with a very large community of people working on it. Currently Talentier uses the 7th version, despite the fact that a more recent version already exists. One of the future plans is to migrate to that version.

Being open source the community is improving the tool. This is divided into two parts: the core (included by default, it is installed with each version of Drupal) and the projects (separate modules that people share so that the whole community can use them).

The core is suffering security updates regularly; This is one of the strong points of using a framework. Another is the ability to use existing modules to avoid having to start from scratch more complex tasks.

In Talentier, the core of Drupal 7 is used, and at present we are working on about 100 projects of the community, in addition to about 30 own modules. For the realization of the project, one of the modules of the community that helps to carry out API was used. Currently there are 5 different projects that help the development of API.

Another of the advantages of Drupal is the management of the content created on the platform, since it makes it modular and updatable. We can also highlight its characteristics in terms of control and management of users and roles or routes and aliases.

Talent Clue also uses Drupal 7.
MySQL is the database technology that uses both Talentier and Talent Clue, since it is one of those supported by Drupal.

This database management system follows a relational model, based on different relationships. These relationships could be considered logically as data sets called tuples. Each relationship works as a table composed of records (rows) and fields (columns).

The main advantages of relational databases are that they avoid duplication of tables and that they guarantee referential integrity. In other words, if a record is deleted from a table it is deleted in all other dependent records.

The main disadvantage compared to non-relationship models is the performance if the base is of a considerable size.

To use MySQL in Drupal, there are own methods to make queries, insertions or modifications. All of them are duly exposed in the Drupal documentation.
5.1.3 HTML/CSS/JavaScript/JQuery

In addition to PHP, common web development technologies have been used, since Talentier is a web tool.

The use of these technologies is constant in the development with Drupal, but the main use was done at the moment of replacing the existing pop-up at the beginning of the project with the current one.

Later, with the explanation of the development of this part, we go into detail about the technology that was used at each moment. HTML is the technology used to define the elements; JavaScript and JQuery establish the ways of acting of the elements defined by HTML, and CSS is the tool to give styles to those elements.
5.1.4 JSON

It is the technology used to make the exchange of data in API requests between platforms (Talentier and other ATS).

JSON, is the acronym for JavaScript Object Notation. It is a light text format for data exchange.

It allows the exchange of numbers (positive, negative or decimals), strings of characters, booleans, null values (null), vectors and objects. Thanks to the sending of this type of data through the API you can make the use that is convenient (either to make queries, create new objects or insert databases). This is possible thanks to the fact that current technologies are capable of translating objects in JSON format.

During the project, to make requests and API translations, cURL has been used. This allows you to add headers and parameters to the requests and then translate the JSON response to PHP arrays.
5.1.5 Git

Git is the technology used by the development team for the version management of the repository. During the development each developer works on a different task but for the same tool. Consequently, the participants do not know exactly what part of the code is modifying the others, and that is why a control mechanism is needed so that in case two developers modify the same part of the code at the same time, the participants are aware of the conflict and can solve it.

Both the Talentier team and Talent Clue use git, but the repositories are hosted in different review services. In the case of Talentier, Github is used and Talent Clue uses BitBucket.
5.2 Software/Hardware used

The hardware used on this project was a 2010 MacBook Pro, with this specs:

![OS X El Capitan](image)

The software used during the development of the project were:

5.2.1 Drush

Drush is a command-line interface of shell for Drupal. Simplify and accelerate the use of Drupal; by means of a series of commands, it allows to develop actions that would be carried out through the navigation of the administrator interface in Drupal.
5.2.2 Pantheon/Platform:

Pantheon and Platform.sh are the two hosting services that have been used in Talentier during the development of the project. Both services offer compatibility with Drupal.

When starting in Talentier the system used was Pantheon, but at the beginning of April the system was changed, since this offered many better features.

Pantheon only had servers in the United States, which made navigability on the web slow. This was the main cause of the change, but it was not the only one, since for example Platform allows managing the repository versions in a few clicks.

Another key benefit of this system is the creation of environments for each Pull Request created from Github, which allows the development team to test the new functionalities without having to download any code to the local machine or having to change the bases of data.
5.2.3 Github + ZenHub

Github is the web service where the project repository was uploaded and ZenHub is an extension of Github that allows you to create tickets and manage them among the development team.

Thanks to the combination of GitHub and ZenHub, the development team was organized daily, allowing the visualization of the SCRUM - Canvas work methodology.

As already explained, the methodology is that the course of the project always flows. The way to represent it is as follows: the tickets (tasks to be done) move from the left column to the right, from Backlog to Closed.

In the Column of Backlog all pending tickets are stored, ordered from top to bottom, from the highest priority to the lowest priority. In the next column (This Week), there would be the tickets expected to be made that week (which would be equivalent to a Sprint in SCRUM).

The development of the tickets is already indicated in the next column (Doing). From this column there is a number that indicates the maximum number of tickets that can be open at the same time to prevent the work from staging at some point. In the case of Doing there can only be 5 tasks at same time in development. This means that until one of the tickets is over, a developer can’t start to develop a new one, so it would have several options to choose from: either check the code of a ticket already developed, or do the testing, or, for last help out in some task where help is needed.

All tickets are designed to be short tasks (less than a day). Therefore, it usually does not happen that the workflow stages at any point.
There is also the Integrate column, which contains those tickets that are part of a larger task. So, they need more than one ticket to be reviewed and tested.

The next part of the flow, once the development of a ticket is finished, we see it in the Review column, where the code is revised (language standards and others) and then moved to Testing. In case of not passing the review, a label will be assigned, such as "Needs more quality" or "Revision". Tickets marked with these labels are modified without moving from the Review column, since tickets can only be moved from left to right so as not to modify the flow. Once the bugs have been corrected, the label is deleted and when the Review is done, they are moved to Testing.

In the Testing column, the ticket is moved next to the PR, since platform.sh creates PR environments, in which the ticket testing can be done. In case of finding an error, a label will be added indicating that it does not work and that revision is needed. Once corrected and passed the testing, it can be moved to the Release column.

In the Release column are all the completed tickets that can be deployed into the platform to which all users have access. Changes are always uploaded early in the morning. Once uploaded the tickets are closed and are registered in the last column on the right (Closed).
5.2.4 SourceTree

It is the software that the development team uses to manage git with a graphical interface. This allows several repositories at the same time, both Github and BitBucket.

It is the visual way to create branches, pushes or pulls of the repository or to solve merge conflicts.

A practical functionality that has is the ability to create stashes, a way to save code modifications without uploading to be able to change branches and be able to recover those changes later.

5.2.5 Docker

Docker is a system to configure secure environments and reproduce the same production conditions for all developers, thus reducing test times and adaptations in the production environment.

Docker is an open source project with which you can create "containers" (light virtual machines). These containers can also be downloaded from a general repository. For example, in Talentier no new one has been created, but previously created containers are used.

The use of Docker was automated by creating scripts that are executed when the computers are started.
5.2.6 MySQL Workbench

It is the software to manage the database with graphical interface. It allows to manage both the Talentier database and the Talent Clue database at the same time. Although it can be modified via the application, it rarely modifies manually.

5.2.7 Sublime Text

It is the text editor used, which allows you to install extensions that facilitate development (such as seeing the changes made in the branch or selecting the language of the file to be able to differentiate colors, classes, objects, strings, etc.).

5.2.8 iTerm

Terminal software that facilitates operation with various shortcuts and styles.
5.2.9 Slack

Communication software used in Talentier to speak the whole team, through different work groups (Marketing, Dev, etc.) as well as one-to-one communication with each member of the company.

In addition, this service allows you to integrate triggers on the web that gives you notifications, such as when a user registers or creates a job offer.

Slack also allows you to use addons, such as the creation of video calls through Appear.
5.2.10 Postman

Postman is used to test API endpoints and help document it. With this software you can test any type of request (GET, POST, PUT, PATCH, DELETE). It also allows to add the headers and POST parameters that are desired.

Example of an endpoint view.
6.1 Phase I Development

The development of the project was carried out in a Mac environment. To start
the project, environments had to be prepared, both Talentier’s and Talent Clue’s,
because the team of Talentier has access to the Talent Clue repository to make
modifications in the integration with the other platform.

It’s important to know that were used 4 development environments:

- **LIVE**: Production environment, where users/clients have access on the
  web.

- **DEV**: The environment where all the new features are tested before
  deploy it to production.

- **TEST**: This environment only is used by Talentier, and is used to do
  some demos to the clients/recruiters.

The 3 environments are configured on the same server, those environments
are provided by the same service.

- **LOCAL**: This is the environment where every developer work on and
  had to be configured at the beginning of the project.
6.1.1 Talentier’s Environment

To prepare the Talentier environment first of all drush must be installed, create a new folder and download Drupal 7 with the command:

```bash
drush dl drupal-7.x
```

Then, with the Makefile file of the Talentier project, that contains all the configurations and packages needed, the drush command has to be executed:

```bash
drush make --no-core talentier.make
```

With these commands it is possible to generate the code of the Drupal own modules and the community projects. To add the custom modules you have to download the Github repository. Before, however, you have to download the Source Tree software, which is the Git system that has been used for the development of the project. Once downloaded, the repository must be synchronized in the /sites/default folder.

Next, docker must be installed to configure the environments of the local machine to match those of the production environment.

Once docker is installed, we copy the file fig.talentier.yml (file that contains the configuration of the environment) also in /sites/default.

To start environments, it is necessary to create a script, because each time the computer is turned on, the environment must be started. The content of this script is:

```bash
#!/bin/bash
cd Talentier/httpdocs/sites/default
docker-machine start tt
eval $(docker-machine env tt)
docker-compose -p tt -f fig.talentier.yml up -d
docker run --add-host local.talentclue:192.168.99.100 referup/apache cat /etc/hosts
```
The next step is to create the database in the environment created by docker. For that first is needed to enter in the mysql container, create the database and copy the production database. Before this you must copy the file talentier.sql in the docker /data/mysql/master folder. Next, you have to enter into the mysql container with the command:

docker exec -it tt_mysql_1 bash

Once inside needs to be created, by the commands:

TERM=linux mysql -uroot -p
create database talentier;
exit;
TERM=linux mysql -uroot -p talentier < talentier.sql

To finish the environment installation, in the hosts file must to be added the reference:

192.168.99.101 local.talentier

6.1.2 Talent Clue’s Environment

To add the Talent Clue environment, the same steps have to be followed, but with the Talent Clue own files, modules, docker and database.

6.1.3 Drupal

Previously I had never worked with Drupal, so I had to dedicate several hours of learning, for the learning I used the Drupal documentation and a book provided by the director, as well as having the support of the director for any questions.

During this learning, we studied how to create users or add routes to how to edit community modules or create my own modules.
Once the preliminary study of Drupal that lasted about two weeks was done, I began to navigate into the menus of Talentier and Talent Clue platforms, also to make the first look at the code of these. A few days later, the first simple tasks began, such as modifying forms or fields of user profiles. Another week later the development work started.

As we mentioned earlier, Drupal has a high learning curve. For this reason, it is best to start from the basics and move forward little by little before entering content creation.
6.2 Phase II Development

This phase was divided into 6 parts, all of them defined by the GEP subject. The realization of this was to serve to make a first analysis of time, costs and a first overview of the project, which would help the final phase of presentation of the project, since it could make a real comparison of what was planned and what actually happened.

GEP was divided in the next deliveries:

- 1\textsuperscript{st} delivery: Definition of scope and contextualization.
- 2\textsuperscript{nd} delivery: Temporal planning.
- 3\textsuperscript{rd} delivery: Economic management.
- 4\textsuperscript{th} delivery: presentation preview.
- 5\textsuperscript{th} delivery: Speciality.
- 6\textsuperscript{th} delivery: Delivery and presentation.
6.3 Phase III Development

The third phase was divided into two different parts. The first was focused on the current study of the system, the mechanisms that were used to carry out the integration with Talent Clue and what had to be the result of the project and for what purpose. Once the study was finished, we had to find the best solution and define step by step what actions had to be carried out during the implementation phase.

To understand the purpose of API functionality it is important to understand the process when outsourcing a candidate search with Talentier through an ATS. In the platform of the ATS a job offer is created, once created, if you want to search for candidates through Talentier you must export the offer to Talentier, so that the recruiters can see all the necessary conditions and requirements. Once the recruiters send resumes of candidates, it is important to know the status of these, if they have been discarded as candidates or if they have been considered as selected, for this another feature is needed to know the status of these. Finally if you want to hire one of the candidates proposed by a recruiter, it is necessary to know what the salary will be to know the percentage that the recruiter takes, so you also have to export the offer so that the recruiter can accept or not the conditions from Talentier.

Once the outsourcing process is described, it is possible to go into more detail about the points to be taken into account during the study.

6.3.1 Study

To start the study phase, a meeting was held with the director, who was in charge of implementing the previous version of the Talentier communication with Talent Clue. This meeting served to know how it was done before and where the development had to go.

There are three main resources during the outsourcing process on which the API is focused, these are the Jobs (job offers that are published), CV (CVs sent by
the recruiters) and CP (Candidate Placement, proposal to hire a company to a recruiter candidate).

In the previous integration, two Drupal modules were used (Services and Migrate), one of the problems was that Migrate was configured to only work with the Talent Clue connection, which did not make it scalable with other platforms. On the other hand, the connection could be simplified using a single module.

The study began looking Drupal modules that will help in the development of APIs, the different modules found were:

- Services: The one used before, the most used by the Drupal community. Uses the SOAP protocol to answer the requests, where the responses are longer than REST. This was the module with more documentation.

- RestWS: This module was underused by the community and was not updated often. Also didn’t have a good documentation.

- Endpoint: Few functionalities and documentation, for example the documentation didn’t explain how to manage the authentication.

- RESTful: The chosen decision. Because it has a good documentation, could be used for any platform and good performance thanks of the REST architecture. Another good point was that it had updates often.

It should be noted that Drupal 8 already has REST support and documentation for custom APIs.

Once the module that was going to be used was decided, all the functions that had previously been new API that could be incorporated were listed. After listing the functionalities and having the resources clear, it was possible to move on to the design phase of the API.

Regarding the usability problems of the popup, they were also listed to take them into account and design a solution. The problems seen were; the size, since it was only a relatively small central box. the appearance of Talentier’s navbar, this took away the rest of the popup. It was not fixed, that is, if you scrolled down then you could continue scrolling on the background screen and finally sometimes you did
not see the close button, since it was small and sometimes you did not identify enough.

6.3.2 Design

At the beginning of this task, a meeting was again held with the director to verify that the idea thought was good for the Talentier team. During the study many features that could be added, how to obtain lists of recruiters, lists of recruiters filtered by countries in which they work or sectors that dominate were contemplated. Or lists of positions also with filters and many more. But it was decided to focus on the fact that it was functional for integration with other ATS and that it could be updated in the future.

The first task before defining the endpoints was to decide what type of authentication would be needed to use the API. As the RESTful module offered the possibility of using basic authentication (username + password) or session token, which in turn requires basic authentication, it was decided to use the token, since it was a bit safer.

Just after the authentication point was clear, the API endpoints to be designed were defined:

Job resource endpoints:

• Create a Job:
  - Justification: Endpoint used to export some job offer from an ATS to Talentier, to allow the recruiters accept or reject the proposal.
  - Type: POST. To create a new Job object with all the information.
  - Input: Job data object.
  - Output: Job Id on Talentier’s side.
  - Authentication: Token.

• Update a Job:
If some ATS needs to modify some information like salary, deadline, etc.

Type: PATCH. To modify an existing object.
- Input: Job Id plus the data to modify.
- Output: No.
- Authentication: Token

Get unread messages of a Job:
- Justification: This was an extra feature not seen at the beginning, allows to know if a position has new messages.
- Type: GET. Query that return the number of unread messages.
- Input: Job Id.
- Output: Number of unread messages in a job.
- Authentication: Token.

CV resource endpoints:

Update CV’s status:
- Justification: It’s necessary to a recruiter to know the candidates status to know how the process is in every moment.
- Type: PATCH. Modifies the CV status (accepted, rejected, etc.)
- Input: CV Id and the new status.
- Output: No.
- Authentication: Token.

CP resource endpoints:

Create a CP:
- Justification: The Talentier structure stores candidate proposals as a Drupal node, so it has to deal separately with the position. And it serves for the recruiter to accept or reject the offer of the company when hiring one of its candidates.
- Type: POST. Creates a new CP object, with the sent data that contains the proposed salary to the candidate proposed by the recruiter.

- Input: Position Id, CV Id and proposed recruitment data.

- Output: CP Id on Talentier’s side.

- Authentication: Token.

- Update a CP:
  - Justification: If a recruiter rejects the conditions could be changed.
  - Type: PATCH. CP object modification with the new data.
  - Input: CP Id, and recruitment data.
  - Output: No.
  - Authentication: Token.

Only the necessary functionalities to carry out the outsourcing process were designed and developed, all the possible functionalities of an API such as queries for positions created in Talentier, list of available recruiters, etc. they were not contemplated for this project, but it is not ruled out to be added in the future.

Once the endpoints were defined, an Epic with the tickets of what would be the API development was created in ZenHub, defining the details of each task to be performed.
A new design for the new pop-up to correct the usability problems was found. This consisted in creating a custom template that would eliminate the navbar and so that the elements could be treated separately. As well a new bigger iframe that always adjusts to any monitor with the gray background and the button to close out of it.
6.4 Phase IV Development

Once the API was documented and the tickets defined, it was time to start the implementation. For that, I started learning to use Drupal's RESTful module, since it had good documentation.

6.4.1 Module

The installation of the module was done with Drush, this installs all the dependencies with other modules. The module was also added to the Talentier Makefile, so when merges into the repository it will be automatically installed in the production environment.

Once installed this enables a new menu, only accessible in administrator mode, to be able to configure things like the API base path or what kind of requests to accept.

The first thing that was studied about the use of the module was how to create resources and endpoints. For this, the simple fact of creating an endpoint of a resource Job without any type of authentication. After that I started the study of how to manage the authentication.
To authenticate the use of the API, the RESTful module supported basic authentication, token or cookie. To make use of token authentication you have to use an endpoint that offers you by default the RESTful module which needs the basic authentication of a registered user.

### 6.4.2 Implementation

Once the study and test of the module was done, the development of the designed API was started.

The first point of the development was to deal with the authentication solution. Its implementation consisted of three parts; first, create a user role called api, second, create a user named api and third, give the api user access to the API resources: Job, CV and CP.

The solution was simple, but the issue of authentication took much longer than expected, since initially it was not contemplated the possibility of creating a user for the API, and it was thought to return to the design phase to create new endpoints to register new users to the Talentier platform and these have access to their own resources, but in the end the solution was found, which in turn is considered as replicable as possible.

Once the authentication problem was resolved, the development of the endpoints began. The order of the outsourcing process was followed first, the export and modification of a Job, changing the states of the CVs, creation and modification of the CP and finally the extra functionality of the messages.

The realization of the endpoints was simple thanks to the fact that the module allowed creating own resources or using drupal nodes, how the three resources were already part of the Talentier node structure was easier. It was simply to implement the functions and associate them with each method of the endpoint.
With the realization of the extra functionality to see the unread messages, a badge notification was added to the message button with the number of unread messages.

Extra functionality of none read messages

Once the API tickets were finished, the popup reform began, for this, the first job that was done was to create a custom template, this was assigned to any web route that ended with /ttbox, which allowed to treat any element separately, simply checking the last element of the browser path. This supposed changing all the links in the Talent Clue code, a task that was performed at the time of the substitution. As the popup did not allow to return to the previous screen, the back button in the template was also integrated.

These changes did not correct all the usability problems of the popup, but due to the deadline, it was decided to move to the substitution phase and if during the substitution it was time to change more things would be done during that.
6.5 Phase V Development

In the substitution phase, not only the previous communication system was replaced by the API, but also the usability errors of the popup integrated in Talent Clue were completed.

6.5.1 API

The first step that was followed to make use of the API from Talent Clue was to add the API user password in a server configuration file, in order to be able to authenticate via token.

Once the authentication was incorporated, a test of any call in the production environment was carried out. Once connected, was needed to replacing one by one the calls of the previous integration by the calls to the new API. The task of testing these changes was not tested until the sixth phase of the project, since they were part of the last stage of the project.

How Talent Clue uses Drupal, that is PHP chose to use cURL, to make calls to the API. It should be noted that cURL is available in 30 programming languages.
6.5.2 Pop-up

As the replacement of the API did not take much time and the testing and debugging of the sixth phase was going to be short, the last usability flaws found earlier in this phase could be fixed.

To do this, the size of the pop-up was modified making it larger, which meant that the elements were not so compact. In addition, the close button that was previously small and sometimes not very identifiable, moved out of the popup and also increased the size, making it more visible. It was also used to add an effect to know that the popup was loading, since previously it remained white while waiting.
6.6 Phase VI Development

As mentioned with the work methodology, the tickets were tested before being considered finished, once all the tasks where done and tested separately the whole project test and debug process started, with all the browsers and possible versions, since previously developing something else we have problems of compatibility with Internet Explorer.

Testing the part of the API only a small error that was solved quickly occurred, the error was that it did not change the state of the CV well when you wanted to discard it, but it was simply to change the POST parameters that were not correctly.

Finally testing the popup, several errors were found and had to be corrected, such as that once closed it could not be opened until the page was refreshed. But all the errors found were corrected correctly and on time.
6.7 Final Phase

The final phase is not included as development, but it was going to involve a great time of completion, so it has been considered as one more phase of the project to be carried out. Consisted into do the documentation and presentation.
7.1 Economic management

This section will explain how the estimated budget was calculated and the real project price.

To carry out the estimated budget, we started with two premises, trying not to exceed € 10,000 and using as many free tools as possible, in order to reduce costs, since Talentier is a startup that, although it already generates income, is still a small company. It does not consist of much budget.

7.2 Resources identification

The necessary resources were divided in the next:

- **Human resources:**

  The people who was going to do the project.

  - **Developer:** Salary for working with the agreement of the Polytechnic University of Catalonia. Responsible for designing and implementing the API.

- **Offices:**

  Places where the project was carried out, both offices include the costs of electricity and internet connection.

  - **Talentier'd office:** Place where all the Talentier team works, placed is in La Salle Bonanova.

  - **TalentClue's office:** From the month of April, it was planned to rent two tables in the Talent Clue offices, in case cooperation or meetings with them were needed.
Hardware

All the hardware resources necessary to carry out the project.

- **Laptop:** The main tool to do the project.
- **Server:** The server hosting where the Talentier’s website was hosted.

Software

- **Drupal:** Framework used to develop the project.
- **Drush:** Command Shell for Drupal.
- **GitHub:** Repository service.
- **ZenHub:** GitHub’s extension.
- **Source Tree:** Interface Software used to manage Git.
- **MySQL Workbench:** Interface software to manage the Database
- **Sublime:** Text editor software.
- **iTerm:** Terminal software used.
- **Slack:** Communication software used.
- **Postman:** API usage software
- **Docker:** Environment software.
● Unforeseen:

It would be the costs not taken into account at the beginning, the estimated budget for the unforeseen is the difference of the total minus the maximum budget established at about € 10,000.

7.3 Cost estimation

● Human resources:

  − **Developer salary**: According to the agreement made between Talentier and the Polytechnic University of Catalonia, this will charge € 960 per month.

● Offices:

  − **Talentier's office**: The office cost has a price of 800€ per month.

  − **Talent Clue's office**: The desk has a price of 150€ per month since April.

● Hardware:

  - **Laptop**: Personal MacBook with no additional cost.

  - **Server**: The Pantheon hosting price was 100€ every month.

● Software:

All the software used is free, Sublime has a paid version.

In the following table you can see the estimated cost of the project:
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Human resources</strong></td>
<td><strong>Salary</strong> 960€ (5 months)</td>
<td></td>
</tr>
<tr>
<td><strong>Offices</strong></td>
<td><strong>Talentier 800€ (5 months) + Talent Clue 150€ (3 months)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Hardware</strong></td>
<td><strong>Server 100€ (5 months)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Software</strong></td>
<td><strong>0€</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>9.750€</strong></td>
<td></td>
</tr>
</tbody>
</table>

250€ for unforeseen.
7.4 Project cost

Once the project has been completed, it has been possible to calculate the total cost of the project. This has been break down month by month. The only unforeseen event not calculated at the beginning of the project was that the server changed, but this turned out to be cheaper. So the total price was lower than initially thought.

**February:**

<table>
<thead>
<tr>
<th>Human resources</th>
<th>Salary</th>
<th>960€</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offices</td>
<td>Talentier</td>
<td>800€</td>
</tr>
<tr>
<td>Hardware</td>
<td>Server</td>
<td>100€</td>
</tr>
<tr>
<td>Software</td>
<td></td>
<td>0€</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td>1.860€</td>
</tr>
</tbody>
</table>

**March:**

<table>
<thead>
<tr>
<th>Human resources</th>
<th>Salary</th>
<th>960€</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offices</td>
<td>Talentier</td>
<td>800€</td>
</tr>
<tr>
<td>Hardware</td>
<td>Server</td>
<td>100€</td>
</tr>
<tr>
<td>Software</td>
<td></td>
<td>0€</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td>1.860€</td>
</tr>
</tbody>
</table>

**April:** In the month of April, the server was changed. But during this month the license of both months was paid in case it was necessary to return to the previous service. In addition, Talent Clue was started every two weeks, and two tables were rented, one for the director and one for the student.

<table>
<thead>
<tr>
<th>Human resources</th>
<th>Salary</th>
<th>960€</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offices</td>
<td>Talentier</td>
<td>800€</td>
</tr>
<tr>
<td>Hardware</td>
<td>Servers</td>
<td>140€</td>
</tr>
<tr>
<td>Software</td>
<td></td>
<td>0€</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td>2.050€</td>
</tr>
</tbody>
</table>
May: How the server gave very good performance results and offered us very useful new features was decided to unsubscribe from the other hosting service.

<table>
<thead>
<tr>
<th>Human resources</th>
<th>Salary 960€</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offices</td>
<td>Talentier 800€ + Talent Clue 150€</td>
</tr>
<tr>
<td>Hardware</td>
<td>Servers 40€</td>
</tr>
<tr>
<td>Software</td>
<td>0€</td>
</tr>
<tr>
<td>Total</td>
<td>1.950€</td>
</tr>
</tbody>
</table>

June:

<table>
<thead>
<tr>
<th>Human resources</th>
<th>Salary 960€</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offices</td>
<td>Talentier 800€ + Talent Clue 150€</td>
</tr>
<tr>
<td>Hardware</td>
<td>Servers 40€</td>
</tr>
<tr>
<td>Software</td>
<td>0€</td>
</tr>
<tr>
<td>Total</td>
<td>1.950€</td>
</tr>
</tbody>
</table>

In the end, the two premises on which the project's budget was based, the maximum use of free software and not exceeding € 10,000 could be achieved. The total cost of the project break down month by month is as follows:

<table>
<thead>
<tr>
<th>Total</th>
<th>February</th>
<th>1.860€</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>March</td>
<td>1.860€</td>
</tr>
<tr>
<td></td>
<td>April</td>
<td>2.050€</td>
</tr>
<tr>
<td></td>
<td>May</td>
<td>1.950€</td>
</tr>
<tr>
<td></td>
<td>June</td>
<td>1.950€</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>9.670€</td>
</tr>
</tbody>
</table>
7.5 Economic viability

This project started from the economic need of the Talentier company, due to the fact that part of the income comes from users who outsource their selection processes with Talent Clue, allowing them to join more ATS in the future, thus increasing the number of users, which meant greater number of income.

If Talentier manages to integrate into any new ATS, the price paid for the development of this project can be amortized in a short period of time, since once the project is completed it will not mean more economic cost.

On the other hand, the cost of this project is not very high, taking into account that most of the budget would be paid anyway if this project is not carried out, so it can be considered economically viable.
8.1 Sustainability matrix

The next matrix shows the impact of the project on sustainability:

<table>
<thead>
<tr>
<th></th>
<th>PPP</th>
<th>Useful life</th>
<th>Risks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environmental</td>
<td>Design consumption</td>
<td>Ecological footprint</td>
<td>Environment risks</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>18</td>
<td>-3</td>
</tr>
<tr>
<td>Economic</td>
<td>Bill</td>
<td>Viability plan</td>
<td>Economic risks</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>15</td>
<td>-3</td>
</tr>
<tr>
<td>Social</td>
<td>Personal impact</td>
<td>Social impact</td>
<td>Social risks</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>Sustainability range</td>
<td>11</td>
<td>38</td>
<td>-6</td>
</tr>
<tr>
<td></td>
<td>43</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8.2 Environmental sustainability

The environmental impact in terms of energy cost has been the electricity consumption produced by the development team, the hosting server and the one produced in the offices, taking into account that the computer is not a high-performance computer and the hours of work were in daytime has not been an excessive expense. As for the server hosting service has not been able to know the consumption, but taking into account that it is always running even after the project can be considered a bit expensive energy.

Regarding the previous manufacture of the used development equipment, it is known that it produces waste that affects the environment, since it was only a team it can’t be considered a great environmental impact, besides it is not available only for the realization of the project, the team is already 6 years old. When changing the equipment, the best way to have the least impact on the environment.
Being a software project this does not produce a direct environmental cost, in addition to one of the key points of the project is that it is replicable, so the number of users using the API will not directly affect the environment.

### 8.3 Economic sustainability

The cost of the project can’t be considered expensive, since the majority of the funds would have been consumed equally if this project had not been carried out, since the student who started working at Talentier didn’t enter to work to carry out this project, but it was later proposed because of the latter's need to complete a final degree project.

Since Talentier is a small company with less than one year of life and at the moment does not count on large revenues, it has the mentality to save as much as possible, which is why all the software used during the project was either free or open source.

On the other hand, the API performed has been made for very specific functionalities, so this could be expanded in the future, if you want to invest in improving it with new developers could mean more economic cost to the project.

### 8.4 Social sustainability

This project is based on integrate Talentier in any ATS, this means helping companies that publish job offers through some ATS, to find better candidates for the proposed positions. Another favorable point for users (companies) using Talentier through the API is that it is free of charge, since Talentier charges a percentage to recruiters, not to companies. And the ATS that integrate it would generate income, since Talent Clue also charges a percentage of Talentier's benefits if the job offer came from Talent Clue.

Finally, the author of the project the realization of this has meant a great first work experience in software development, in addition to providing great knowledge in several different technologies, which will serve you for the future.
9. References

<https://www.drupal.org/documentation>  
[Consulted on January 22nd of 2016]

Pro Drupal 7 Development - Todd Tomlison & John VanDyk  
[Consulted on January 25th of 2016]

[3] SCRUM - SCRUM’s methodology [Book]  
Agile Project Management with SCRUM - Ken Schwaber  
[Consulted on January 16th of 2016]

<http://www.tomsplanner.es/>  
[Consulted on March 6th of 2016]  
[Consulted on July 14th of 2016]

<https://github.com/RESTful-Drupal/restful/wiki>  
[First consultation on April 6th of 2016]

Personal Kanban in a Nutshell - Jurgen De Smet  
[Consulted on May 4th of 2016]

<http://www.w3schools.com/>  
[Consulted since the start of the development]
10.1 A annexed: Talentier’s API Documentation

Default endpoints:

- Get the access-token:

  Request URI: https://talentier.com/api/login-token
  Request Method: GET
  Content-Type: application/json
  Authorization: Basic

Job endpoints:

- Create new job:

  Request URI: https://talentier.com/api/v1.0/Job/new
  Request Method: POST
  Content-Type: application/json
  acces-token: Token
  Input Data: {
    "title": "job_title",
    "description": "job_description",
    "location": "job_location",
    "industry": "job_industry",
    "sector": "job_sector",
    "salary_min": "job_salary_min",
    "salary_max": "job_salary_max",
    "vacancies": "job_vacancies"
  }

- Update a job:

  Request URI: https://talentier.com/api/v1.0/Job/update
  Request Method: PATCH
  Content-Type: application/json
  acces-token: Token
  Input Data: {
    "id": "job_id",
    "title": "job_title",
    "description": "job_description",
    "location": "job_location",
    "industry": "job_industry",
    "sector": "job_sector",
    "salary_min": "job_salary_min",
    "salary_max": "job_salary_max",
    "vacancies": "job_vacancies"
  }

- Get unread messages of a job:

  Request URI: https://talentier.com/api/v1.0/Job/unread_messages
  Request Method: GET
  Content-Type: application/json
  acces-token: Token
  Input Data: {
    "id": "job_id"
  }
CV endpoints:

- **Update CV status:**

  Request URI: https://talentier.com/api/v1.0/CV/update
  Request Method: PATCH
  Content-Type: application/json
  acces-token: Token
  Input Data: `{"id":"cv_id", "status":"cv_status"}`

CP endpoints:

- **Create new CP:**

  Request URI: https://talentier.com/api/v1.0/CP/new
  Request Method: POST
  Content-Type: application/json
  acces-token: Token
  Input Data: `{"jid":"job_id", "cvid":"cv_id", "salary_annual":"cp_salary_annual", "salary_var":"cp_salary_var"}`

- **Update a CP:**

  Request URI: https://talentier.com/api/v1.0/CP/update
  Request Method: POST
  Content-Type: application/json
  acces-token: Token
  Input Data: `{"id":"cv_id", "salary_annual":"cp_salary_annual", "salary_var":"cp_salary_var"}`
### API codes:

<table>
<thead>
<tr>
<th>Code</th>
<th>Text</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
<td>Success!</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
<td>The request was invalid or cannot be otherwise served</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized</td>
<td>Authentication credentials were missing or incorrect</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden</td>
<td>The request is understood, but it has been refused or access is not allowed</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
<td>The URI requested is invalid or the resource requested, such as a user, does not exists</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error</td>
<td>Something is broken</td>
</tr>
</tbody>
</table>
10.2 B Annexed: Scrum + Kanban Methodology

Rules

Daily Standup

At 9h30, keep it SHORT (15min), FOCUS on creation of VALUE + REMOVING BLOCKAGES
Coordinator: SCRUM MASTER

1. UPDATE metrics and graphs
   a. By scrum master, before meeting
2. OBSERVE and ANALYZE metrics and graphs
   a. Why certain changes in velocity, queue volume, etc?
   b. What are the current blockers? Where is the flow obstructed?
   c. What needs to be done to unblock flow?
3. ORGANIZE the day
   a. What are the current priorities?
   b. Who needs help with certain issues?
   c. Who will help out with them, and when?
   d. Who depends on who for the development of certain tickets?
4. COMMIT
   a. Express personal commitments, so as a team we know we can count on you

Release Plan

1. Prepare RELEASE - every morning (between 8h and 9h15)
   a. If there is a dependency with TC, remove the ‘Closes’ tag in ticket description, so that ticket will remain open until the TC depending part has been deployed
   b. Merge the PRs into development which are in Release column
   c. If there is a MERGE CONFLICT
      i. Move PR to Release column and mark Merge Conflict
   d. Create snapshot of master branch in Platform.sh
2. Do RELEASE - every morning (between 8h and 9h15)
   a. git fetch github
   b. pull github development
   c. push platform master
3. Finish RELEASE - every morning (between 8h and 9h15)
   a. Clean up Release column (if any ticket left over and not closed automatically)
   b. Add released total number of Story Points to DEV Metrics
   c. Pull code & database to TEST and to DEV
4. Tag WEEKLY Code Change - every monday morning (just after release)
   a. Create a PR in github (development into master)
   b. Merge PR
   c. Create tag with number v1.x.[weeknumber]
5 Simple Flow Rules

1. Always **FINISH** the task or micro-task you are currently **FOCUSSED ON**

2. If you are **STUCK**, **LABEL** your ticket
   a. **ORK**: estimation will not be met, important unexpected problems have arisen
   b. **DRAGON**: completely lost in problems, you don’t know what to do next anymore
   c. Communicate your status: right now through Slack, and during next daily

3. If you are **READY** with what you were doing, **DO** next item using the following **RULES**
   a. Take a **BUG** ticket if there is one
   b. **TEST** tickets if there are any (not your own tickets), add labels
   c. **FIX** / **IMPROVE QUALITY** of your code if a ticket of yours is **not working**, **needs changes**, or **needs more quality**
   d. Give **SUPPORT** on any **ORK** or **DRAGON** ticket
   e. PULL FIRST ticket of the **BACKLOG**

4. Except, if there is a **HIGH PRIORITY** ticket on your name
   a. Immediately **STOP** what you are currently doing
   b. Coordinate with the team, who takes responsibility to solve the **ISSUE**
   c. Communicate any problems you are facing, do not post-pone asking for help

5. **RELEASE** will be done, if there is code to be released, **DAILY** between 8h and 9h15

---

**Daily Schedule**

- 8h - 9h15 Release
- 9h30 - 9h45 Daily Standup
- 9h45 - 10h Tech Support
- 10h - 12h30 Development
- 12h30 - 13h30 Review MS (if anything to review), and Tech Support (on demand)

**LUNCH**

- 14h30 -> afternoon Fix code, and finish your stuff

(**Taking always into account, the agenda of other Devs, so if you want your ticket to be retested, finish it asap.**)

---
Definition of Ready

1. DEFINED
   a. New features
      i. Why? Why important
      ii. Who? Who is the user, who will this deliver value to
      iii. What? Expected outcome
         1. Short description
         2. Mock-up or sketch
         3. Design (if new feature and special design needed)
   b. Bugs (what to do to reproduce?)
      i. URL + Screenshot
      ii. Expected result

2. ANALYZED
   a. Split in very small chunks of independent working functionalities
   b. Implementation Plan defined
      i. Cleared out all technical doubts and questions

3. ESTIMATED
   a. Estimations agreed on with whole team (estimation = READY)
   b. No more than 5 story points per ticket

4. CLEAR
   a. Developers understand expected outcome

5. LABEL a ticket if NOT READY
   a. If a ticket is not ready, because something is missing, assign the appropriate label (blue labels are good candidates)
Definition of Done

1. WORKING as DEFINED
   a. Tested and approved (by -internal- client if relevant, during development process)
   b. Tested and approved (by another developer)

2. QUALITY ASSURED
   a. Coding standards respected
   b. Readable and extendable
   c. No manual database changes needed (or very very exceptionally, using label)

3. MERGED
   a. Into github development branch
   b. No dependencies on other tickets

4. RELEASED
   a. Translations, and other manual changes, executed
   b. Available in LIVE
   c. Available in TEST & DEV

LABELS

1. AREA of DEVELOPMENT
   a. THEMING
   b. jQuery / AJAX
   c. TC (TalentClue)
   d. API

2. NOT READY status
   a. needs design
      i. A visual design is needed for this ticket
   b. needs discussion
      i. A discussion is needed between stakeholders, to take decisions
   c. needs refinement
      i. More details are needed, by stakeholders

3. FLOW status
   a. waiting
      i. Depending on external input or action
   b. needs more quality
      i. Code Review not passed, code needs quality improvements
   c. needs changes
      i. Code Review not passed, code needs structural changes
   d. not working
      i. Testing not passed, not working as defined

4. SPECIAL ATTENTION FLOW status
a. **BUG**
b. **HIGH Priority**
c. * ORK *
d. * DRAGON *

5. DEVELOPERS labels
   a. IG
   b. VK
   c. MS

6. CLOSING reason
   a. In progress
   b. INVALID
   c. DUPLICATE
   d. CAN’T REPLICATE

**SCRUM Meetings**

1. **Daily StandUp**
   a. When? EVERY DAY at 9h30
   b. 15min max
   c. Objective:
      i. Focus on Value and removal of blockages
      ii. Organize and coordinate the day
   d. Coordinator: Scrum Master

2. **Refinement Meeting** (grooming)
   a. When? Every TUESDAY, fixed time, 1h
   b. Backlog items that have no estimations
   c. Objective:
      i. Split work in small parts
      ii. Discuss technical issues
      iii. Create implementation plan
      iv. Estimate
   d. Coordinator: Team Lead

3. **Retrospective Meeting**
   a. When? Every two weeks (FRIDAY), 1.5h
   b. Objective:
      i. Learn from past 2 weeks
      ii. Share and propose improvements
      iii. Concrete proposals for change
   c. Coordinator: Scrum Master