Integrative Project of the 4th Semester
System Specification

Licenciatura em Engenharia Informática, ISEP - 2023/2024

Porto, February 26, 2024

Alexandre Bragança (ATB)
Paulo Sousa (PAG)
André Moreira (ASC)
Luis Nogueira (LMN)
Ana Madureira (AMD)

`Version 8, 2024-02-22`

Revision History

Revision	Date	Author(s)	Description
1	2024-02-14	atb	initial version
2	2024-02-16	atb	adds more explicit connection between US and "UC Unidades Curriculares"
3	2024-02-16	pag, asc, lmn	requirements refinements
4	2024-02-20	pag, atb, lmn	requirements refinements
5	2024-02-20	asc	adds details do NFR11
6	2024-02-21	pag	fixes some issues in the description of the application process
7	2024-02-22	atb	minor update
8	2024-02-22	asc	fixes minor typos

Contents

List of Figures

Chapter 1

Context

In the 2023-2024 academic year, the fourth semester (i.e. 2nd year, 2nd semester) of the Degree in Informatics Engineering (LEI) of the Instituto Superior de Engenharia do Porto (ISEP) adopts a teaching-learning process based on the development of a single project that enhances the integration and application of knowledge, skills and competencies of all course units taught through the semester: Applications Engineering (EAPLI), Laboratory and Project IV (LAPR4), Languages and Programming (LPROG), Computer Networks (RCOMP) and Computer Systems (SCOMP). The project, common to all course units, consists of developing the system described in this document in accordance with the general procedures described earlier in the document (written in Portuguese) with the same designation, whose subtitle is "Descrição de Funcionamento".

Chapter

2 System Description

Jobs4U is a company specialized in talent acquisition. The company provides recruitment services for job positions in its clients. The aim of this project is to develop, in an exploratory way, a solution that allows automating the main activities of the company. Therefore, a minimum viable product should be developed in 3 months.

The company's clients are other companies or entities that need to recruit human resources. In response to requests from its clients, Jobs4U develops all activities that allow it to select a set of candidates for job offers (from its clients). At the end of the process, Jobs4U must deliver to its client an ordered list of candidates for each job offer. The final recruitment decision is the responsibility of the client.

2.1 Use Cases

Figure 2.1 illustrates the main use cases of the project.

The system administrator (Admin) is responsible for managing customer entities as well as the company's employees who are customer managers (Customer Manager). This responsibility involves registering entities as well as assigning different roles to system users (i.e., customer manager, operator). It also includes the customer's registration as a user of the system.

Entities send job offers to Jobs4U. This sending can be done by various means (e.g. email, post, telephone), but the automation of this reception is outside the scope of the system. A customer manager will register job offers for the entities he manages in the backoffice. The customer manager will also manage other aspects of job offers, namely the entire candidate selection process. However, the registration of candidates for job offers is carried out by the operators (using some bots to automate the process).

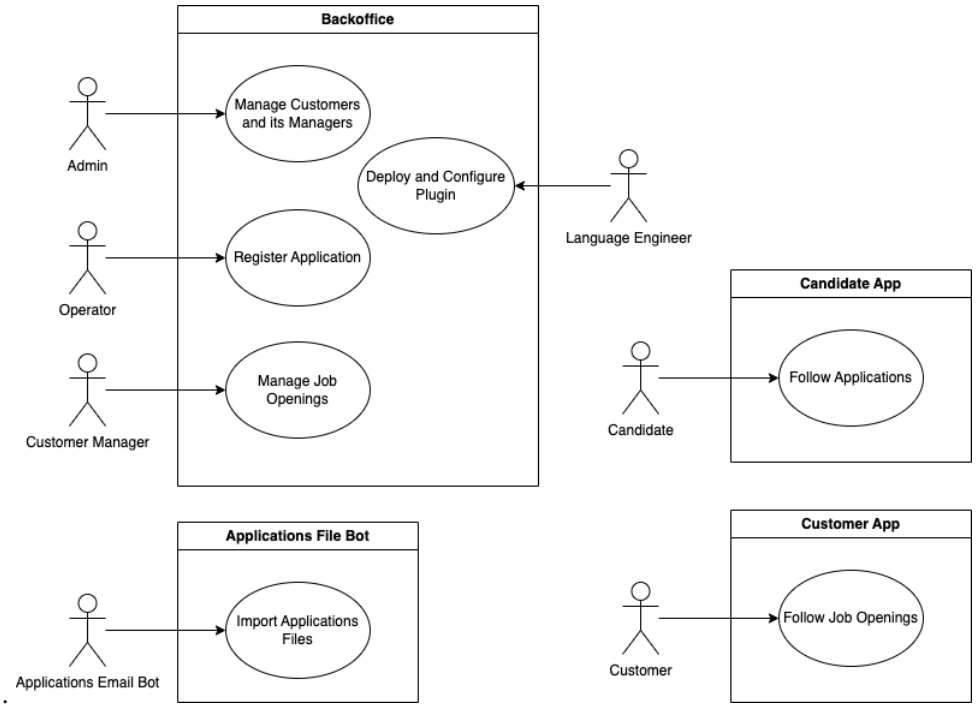
The operator is a company employee whose main responsibility is to monitor the automatic process that registers applications for job offers. These applications are received by email

and processed automatically by the Applications Email Bot (out of scope for this project). The applications Email Bot processes the emails and produces files adapted for integration in the system, that will be automatically processed by the Applications File Bot. The Operator should monitor this process, specially the report files that are produced. The Operator may, eventually, contact candidates if problems exist in their applications. The files produced by the Applications File Bot are used to integrate candidates and applications into the system.

Both candidates and clients have specific applications (console applications) that basically allow them to monitor applications and job offers and receive notifications about them.

The selection of candidates is highly based on automated processes (such as interviews or verification of application requirements) that require the production of plugins (for the Backoffice application) by a Language Engineer. These plugins automate the processing of job interviews and application requirements by applying language processing, that maybe specific for each job opening.

Figure 2.1: Main Use Cases of the Project)



2.2 Functional Details

This sections presents some more details about the project. These details were gathered after an initial conversation with the project owner.

2.2.1 Recruitment Process

The recruitment process for a job opening follows a sequence of phases: application; resume screen; interviews; analysis; result.

- **Application** This is the first phase of the process. During this phase candidates submit their applications.
- **Screening** This phase follows the application phase. In this phase, applications are verified against a set of requirements. Applications that do not meet mandatory requirements are rejected.
- **Interviews** This phase is not mandatory, but is common. During this phase, accepted candidates may be interviewed. Results for the interviews are registered for further analysis.
- **Analysis** During this phase, the applications are analyzed (using all available information like interviews and curriculum) and candidates are ranked.
- **Result** In this phase, candidates as well as customers are notified of the final result.

The customer manager is responsible to setup the process, defining the dates for the phases and if the process includes interviews.

2.2.2 Job Opening

Job openings (or job vacancies) are created in the system by a customer manager. A job opening includes:

- **Job Reference** A unique identifier of the job opening (generated by the system, for instance based on the customer code followed by a sequential number)
- **Title or function** Such as "front end programmer"
- **Contract Type** Such as full-time or part-time
- **Mode** Such as remote, hybrid, onsite
- **Address** Address for job
- **Company** Customer name
- **Number of vacancies**
- **Description**
- **Requirements** The job requirements specification to use for this opening, as described in Section 2.2.4.

2.2.3 Applications

Candidates submit their applications for job openings by email.

There is an Applications Email Bot (outside of scope for this project) that is continuously processing these emails. The Bot processes the emails and produces (in a predefined folder) the following content/files (using the same file prefix for files of the same application):

- A text file with the contents of the email
- A file for each file attached to the email (usually PDF files)
- A text file with the contents of each file attached to the email (processed by an OCR tool)
- A text file with the data of the application and candidate, with at least:
  - job reference
  - email of the candidate
  - name of the candidate
  - phone number of the candidate

There is a second bot application, named Applications File Bot, that processes these files for integration in the system. The Applications File Bot is continuously monitoring the previous referred folder for new applications to be processed. The Bot should copy the files for a shared folder. This shared folder should be organized by job reference (top folders) and then by application (sub folder inside the job reference folder). The Bot should produce a text report of all the processed applications (including applications for job references and files available).

The Operator of the Backoffice will import the files produced by the Applications File Bot and register the applications, creating candidates that dot not exist in the system.

2.2.4 Requirement Specifications and Interview Models

Job vacancies (job openings) must include a **Job Requirement Specification**. This represents a set of application requirements that the applicants must achieve. For instance, we could define a job requirement specification named "front end junior programmer", were candidates must have at least 2 years of experience, a degree in computer science or similar program, and knowledge in, at least, one of the following programming languages: java, javascript, typescript. Usually this information can be collected from the curriculum vitae of the candidate.

Interviews are a very important tool for the evaluation of candidates. Interviews should be based on a set of pre-defined questions. **Interview Models**, as the name implies, are sequences of questions that can be used to register the answers of candidates during interviews.

Both job requirement specifications and interview models follow the same usage. A software engineer, with great competences in language engineering, following directions from customer

managers, designs and implements modules that are dynamically added to the system. These modules contain the implementation of job requirements specification or interview models. These modules should be able to implement the necessary functionalities from processing job requirements or interviews.

A job requirement specification module should:

- Generate a template text file with the requirements to be evaluated and the possible answers for each requirement
- Evaluate if a text file with the requirements for a particular candidate is syntactically correct
- Evaluate a text file with the requirements for a particular candidate and provide the result, approved or rejected, and in case of rejection, include justification

An interview model module should:

- Generate a template text file with the questions to be asked in the interview and the possible answers for each question
- Evaluate if a text file with the questions and answers for a particular candidate interview is syntactically correct
- Evaluate a text file with the questions and answers for a particular candidate interview and provide a numeric grade for that interview

Figure 2.2: Example of a template text file with requirements

```
# Enter the number of years of experience (integer)
Experience-years: 2
# Select one degree (None; Bachelor; Master; PhD)
Academic-degree: None
# Select one or more programming languages you are proficient in (java; javascript; python)
Programming-languages: java, javascript
...
```

Figure 2.2 illustrates an example of a template text file with requirements for a job opening. In this example, the evaluation of this requirements (with the existing values) could result in a rejection since the candidate has no degree. The system should provide a justification, such as "A minimum Bachelor degree is required for the job position.". A similar approach is used for job interviews, but in this case, the goal is not to approve or reject a candidate but to evaluate the answers and calculate a grade for the interview in the range 1-100.

A job interview is a form with a set of questions. Each question as a value associated. The sum of the values for all the questions should be 100. At least the following type of questions should be supported:

- **True/False** A question with only a true or false answer.
- **Short Text Answer** A question with a short text answer. The limit of the answer should the specified by a regular expression.
- **Choice, with Single-Answer** A question with a set of choices where only one can be selected
- **Choice, with Multiple-Answer** A question with a set of choices where many can be selected
- **Integer Number** A question which answer is an integer number
- **Decimal Number** A question which answer is a decimal number
- **Date** A question which answer is a date
- **Time** A question which answer is a time
- **Numeric Scale** A question which answer is one option in a range of integers (ex: 1-5)

According to the answers given in the interview for each question the value of the question is calculate following evaluation rules. For instance, consider the following question, which value is 10:

What programing language should be used for system XPTO?

1. Java
2. C#
3. PHP
4. Javascript
5. Typescript

For this case one could define the following rules used in the evaluation of the grade for the question:

1. If the answer is 4 and 5 then 100%
2. if the answer is 4 then 40%
3. if the answer is 5 then 80%

In this case, if the answer is 1 and 2 the grade is 0% of 10, therefore 0. If the answer is 4, the grade is 40% of 10, therefore 4. The total grade of an interview is the sum of the grade of all the questions.

Chapter

3 Requirements

This chapter presents the requirements for the solution divided in functional requirements and other type of requirements.

☞ VERY IMPORTANT: **All requirements (either functional or non-functional) must follow best practices** for the **software engineering process** (e.g., analysis, design, implementation, tests, deployment) as well as the **implementation/-coding (e.g., code quality)**.

3.1 Functional Requirements

3.1.1 Sprint A

- **G001** As Project Manager, I want the team to follow the technical constraints and concerns of the project
  - **Priority:** 1
  - **References:** These constraints and concerns are described in Section 3.2
- **G002** As Project Manager, I want the team to use the defined project repository (i.e., GitHub) and setup a tool for project management.
  - **Priority:** 1
  - **References:** N/A
- **G003** As Project Manager, I want the team to configure the project structure to facilitate / accelerate the development of upcoming user stories.
  - **Priority:** 1

  - **References:** Define the structure of the project to support the envisioned architecture, such as presented in Chapter 4, including support for adopted technologies (e.g., ANTLR)

- **G004** As Project Manager, I want the team to setup a continuous integration server.
  - **Priority:** 1
  - **References:** GitHub Actions/Workflows should be used
- **G005** As Project Manager, I want the team to add to the project the necessary scripts, so that build/executions/deployments/... can be executed effortlessly.
  - **Priority:** 1
  - **References:** Include scripts for all the major tasks and execution of applications
- **G006** As Project Manager, I want the team to elaborate a Domain Model using DDD.
  - **Priority:** 1
  - **References:** N/A

3.1.2 Sprint B

- **G007** As a Project Manager, I want the system to support and apply authentication and authorization for all its users and functionalities.
  - **Priority:** 1
  - **References:** N/A
- **1000** As Administrator, I want to be able to register, disable/enable, and list users of the backoffice.
  - **Priority:** 1
  - **References:** Alternatively this can be achieved by a bootstrap process
- **1001** As Customer Manager I want to register a customer and that the system automatically creates a user for that customer
  - **Priority:** 1
  - **References:** Alternatively this can be achieved by a bootstrap process
- **1002** As Customer Manager, I want to register a job opening.
  - **Priority:** 1
  - **References:** Alternatively this can be achieved by a bootstrap process
- **1003** As Customer Manager, I want to list job openings.

  - **Priority:** 5
  - **References:** N/A

- **2000a** As Operator, I want to register a candidate and create a corresponding user
  - **Priority:** 1
  - **References:** N/A

- **2000c** As Operator, I want to list all candidates
  - **Priority:** 5
  - **References:** N/A

- **2001** As Product Owner, I want the system to, continuously, process the files produced by the Applications Email Bot, so that they can be imported into the system by initiative of the Operator
  - **Priority:** 1
  - **References:** See NFR12(SCOMP).

- **2002** As Operator, I want to register an application of a candidate for a job opening and import all files received.
  - **Priority:** 1
  - **References:** Import the data from the file that was processed by the Application File Bot in Req 2001. The files should be kept in the shared folder, but the Backoffice application needs to know the references to the file locations.

- **1005** As Customer Manager, I want to list all applications for a job opening.
  - **Priority:** 1
  - **References:** N/A

- **1006** As Customer Manager, I want to display all the personal data of a candidate
  - **Priority:** 1
  - **References:** Just the candidate details

- **1006b** As Customer Manager, I want to display all the personal data of a candidate, including his/her applications.
  - **Priority:** 5
  - **References:** Candidate details + application list

- **1007** As Customer Manager, I want to setup the phases of the process for a job opening.

  - **Priority:** 5
  - **References:** N/A

- **1008** As Language Engineer, I want to deploy and configure a plugin (i.e., Job Requirement Specification or Interview Model) to be used by the system.
  - **Priority:** 1
  - **References:** See NFR09(LPROG).

- **1009** As Customer Manager, I want to select the requirements specification to be used for a job opening.
  - **Priority:** 1
  - **References:** See NFR09(LPROG).

- **1011** As Customer Manager, I want to select the interview model to use for the interviews of a job opening (for their evaluation/grading).
  - **Priority:** 1
  - **References:** See NFR09(LPROG).

- **1012** As Customer Manager, I want to generate and export a template text file to help collect the candidate answers during the interviews.
  - **Priority:** 1
  - **References:** See NFR09(LPROG).

- **2003** As Operator, I want to generate and export a template text file to help collect data fields for candidates of a job opening (so the data is used to verify the requirements of the job opening).
  - **Priority:** 1
  - **References:** See NFR09(LPROG).

3.1.3 Sprint C

- **1013** As Customer Manager, I want to rank the candidates for a job opening.
  - **Priority:** 1
  - **References:** N/A

- **1004** As Customer Manager, I want to edit a job opening.
  - **Priority:** 5
  - **References:** N/A

- **2000b** As Operator, I want to enable/disable a candidate
  - **Priority:** 5
  - **References:** N/A
- **1010** As Customer Manager, I want to open or close phases of the process for a job opening.
  - **Priority:** 1
  - **References:** N/A
- **1014** As Customer Manager, I want to record the time and date for an interview with a candidate.
  - **Priority:** 5
  - **References:** N/A
- **1015** As Customer Manager, I want to execute the process of verification of requirements of applications for a job opening.
  - **Priority:** 1
  - **References:** See NFR09(LPROG).
- **1016** As Customer Manager, I want the system to notify candidates, by email, of the result of the verification process
  - **Priority:** 1
  - **References:** See NFR11(RCOMP).
- **1017** As Customer Manager, I want to upload a text file with the candidate responses for an interview.
  - **Priority:** 1
  - **References:** See NFR09(LPROG).
- **1018** As Customer Manager, I want to execute the process that evaluates (grades) the interviews for a job opening.
  - **Priority:** 1
  - **References:** See NFR09(LPROG).
- **1019** As Customer Manager, I want to get an ordered list of candidates, using the job interview points (grades), to help me analyze the candidates.
  - **Priority:** 1
  - **References:** N/A

- **1020** As Customer Manager, I want to publish the results of the selection of candidates for a job opening, so that candidates and customer are notified by email of the result.
  - **Priority:** 1
  - **References:** See NFR11(RCOMP).
- **1021** As Customer Manager, I want to display all the data of an application.
  - **Priority:** 5
  - **References:** N/A
- **2001b** As Product Owner, I want the system to, continuously, process the files produced by the Applications Email Bot, so that they can be imported into the system by initiative of the Operator
  - **Priority:** 1
  - **References:** See NFR13(SCOMP).
- **2004** As Operator, I want to upload a text file with the data fields (requirements) of a candidate for its verification.
  - **Priority:** 1
  - **References:** See NFR09(LPROG).
- **3000** As Candidate, I want to list all my applications and their state (including the number of applicants).
  - **Priority:** 1
  - **References:** See NFR10(RCOMP) and NFR11(RCOMP).
- **3001** As Candidate, I want to be notified in my application when the state of one of my applications changes.
  - **Priority:** 1
  - **References:** See NFR10(RCOMP) and NFR11(RCOMP).
- **3002** As Customer, I want to list all my job openings, including job reference, position, active since, number of applicants.
  - **Priority:** 1
  - **References:** See NFR10(RCOMP) and NFR11(RCOMP).
- **3003** As Customer, I want to be notified in my application when the state (phase) of my job openings changes.
  - **Priority:** 5

  - **References:** See NFR10(RCOMP) and NFR11(RCOMP).

- **4000** As Customer Manager, when displaying the candidate data, I want the system to display a top 20 of the most referenced words in the files uploaded by a candidate and where they are in the document (ex: file and page or line).
  - **Priority:** 1
  - **References:** See NFR14(SCOMP).

3.2 Other Requirements

This section presents some specific non-functional requirements. This includes some constraints and concerns that should be taken into account when designing and implementing the solution.

- **NFR01 - Programming language** The solution should be implemented using Java as the main language. Other languages can be used in accordance with more specific requirements.
- **NFR02 - Technical Documentation** Project documentation should be always available on the project repository ("docs" folder, markdown format) and, when applicable, in accordance to the UML notation. The development process of every US (e.g.: analysis, design, testing, etc.) must be reported (as part of the documentation).
- **NFR03 - Test-driven development** The team should develop a relevant set of automated tests for every US / Class / Method. The team should aim to adopt a test-driven development approach.
- **NFR04 - Source Control** The source code of the solution as well as all the documentation and related artifacts should be versioned in a GitHub repository to be provided to the students. Only the main (master/main) branch will be used (e.g., as a source for releases)
- **NFR05 - Continuous Integration** The Github repository will provide night builds with publishing of results and metrics.
- **NFR06 - Deployment and Scripts** The repository should include the necessary scripts to build and deploy the solution in a variety of systems (at least Linux and Windows). It should also include a `readme.md` file in the root folder explaining how to build, deploy and execute the solution.
- **NFR07 - Database** By configuration, the system must support that data persistence is done either "in memory" or in a relational database (RDB). Although in-memory

database solutions can be used during development and testing, the solution must include a final deployment where a persistent relational database is used. The system should have the ability to initialize some default data.

- **NFR08 - Authentication and Authorization** The system must support and apply authentication and authorization for all its users and functionalities.
- **NFR09(LPROG) - Requirement Specifications and Interview Models** The support for this functionality must follow specific technical requirements, specified in LPROG. The ANTLR tool should be used (`https://www.antlr.org/`).
- **NFR10(RCOMP)** Functionalities related to the Candidate and Customer Apps and to the Follow Up Server part of the system have very specific technical requirements. It must follow a client-server architecture, where a client application is used to access a server. Communications between these two components must follow specific protocol described in a document from RCOMP ("Application Protocol"). Also, the client applications can not access the relational database, they can only access the server application.
- **NFR11(RCOMP)** The solution should be deployed using several network nodes. It is expected that, at least, the relational database server and the Follow Up Server be deployed in nodes different from localhost, preferably in the cloud. The e-mail notification tasks must be executed in background by the Follow Up Server.
- **NFR12(SCOMP)** The base solution for the upload of files must be implemented following specific technical requirements such as the use of the C programming language with processes, signals and pipes. Specific requirements will be provided in SCOMP.
- **NFR13(SCOMP)** An alternative solution for the upload of files must be implemented following specific technical requirements such as the use of the C programming language with shared memory and semaphores. Specific requirements will be provided in SCOMP.
- **NFR14(SCOMP)** The process to count words of very large files should follow specific technical requirements such as implementing parallelism and concurrency using Java and threads. Specific requirements will be provided in SCOMP.
- **NFR15(LAPR4)** This project has some specific requirements regarding communication and presentation of the project and its results. This is a concern of the project and its related to the presentations for the sprint reviews in the context of the LAPR4 TP classes (i.e., skills module). LAPR4 will provide further specification for this requirement.

Chapter

4 Envisioned Applications

Figure 4.1 presents the envisioned set of applications (i.e., components of the system).

The Backoffice app is used by admins, customer managers and operators. Both the Candidate and Customer apps follow a specific design, as described by NFR10. They connect to a server (Follow Up Server) using a specific protocol (NFR10). Only the server is able to connect directly to the database. It is important to note that the client apps receive notifications originating from the server.

The Applications Email Bot is outside of the scope of this project. It is assumed to exist and to behave as described in Section 2.2.3. The files produced by the Applications Email Bot are to be further processed by the Applications File Bot, which results are to be imported in the Backoffice by the Operator.

The plugins (i.e., Job Requirements Specification and Interview Model) support the language automations described in Section 2.2.4. They are developed by a Language Engineer and deployed and configured in the Backoffice to be used for interviews and requirements verifications.

It is important to note that these are the base applications expected for the solution. Other applications can be included in the solution to meet all requirements (functional and non-functional).

Figure 4.1: Envisioned Applications

.