BABEȘ-BOLYAI UNIVERSITY

Faculty of Mathematics and Computer Science

Specialisation Computer Science

Project Jobbble

Software Requirements Specification

version 1.0.0

author
Alexandru Stoica

February 17, 2019

Contents

Chapter 1

Introduction

1.1 Purpose

The purpose of this document is to present a detail description of our platform Jobbble; description concentrated around features and their use in our system. This document is intended for our developers, stakeholders, designers and quality operators.

1.2 Product Scope

Jobbble represents a platform for listing job and opportunities for students in our local or remote area. In contrast to our competitors, our product offers a gamified user experience, in which students earn points for different activities such as applying for a job opportunity, uploading a resume to their project and so on.

Depending on the number of points a student gets, he will gain more visibility and direct opportunities such as a full-time position within a company.

Companies and start-ups can provide internships using our mobile application and track their applicants for each posted job. An administrator from their human resources department can select an applicant and schedule an interview from our list of candidates. Ones chosen for an interview, our system notifies the student about his progress at a given opportunity.

Furthermore, our system requires a constant internet connection to fetch our data. A database, located on our server maintains the system information.

1.3 Glossary

Definition 1.3.1. *A **user** represents someone which interacts with our system using our mobile application.*

Definition 1.3.2. *A **human resources administrator** represents a company's employee with permissions to create, update and delete jobs from our platform.*

Definition 1.3.3. *A **job** is an activity, often regular and often performed in exchange for payment.*

Definition 1.3.4. *A **database** is a collection of information monitored by our system.*

Definition 1.3.5. *A **stakeholder** represents any person with an interest in our project, who is not a developer, designer or tester.*

Definition 1.3.6. *A **student** is primarily a person enrolled in an educational institution, who attends classes in a course to attain the appropriate level of mastery.*

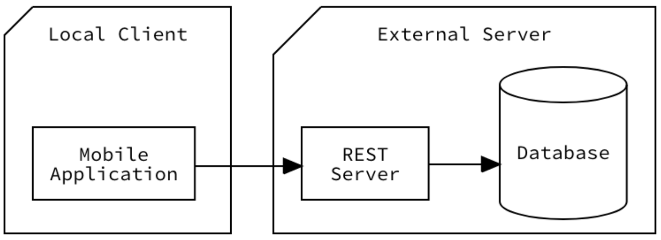
d

Chapter 2

Overall Description

This section will describe the components of our product, our architecture, context and how each component communicates within our system. It also represents our product's primary functions and user's roles. At the end of this chapter, we will focus on a list of constraints and assumptions for our system.

2.1 Product Perspective



Our system will consist of two parts: one or multiple local clients and an external server. Our local client represented by a mobile iOS application will communicate with our external server. Our server will provide management over our database data and will act as a proxy between our local client and our database.

Besides, since our product requires data persistence, we will use a relational SQL database to keep track of jobs and users. The communication between our database and server will be established using Spring. All communications between our local client and the external server will require an internet connection.

2.2 Product Functions

A given user will be able to search through local and external jobs, listed in our database using search criteria such as job's name or job's location. Human resources admins will also be able to search through our database of students and message them directly.

A student will be able to apply for one of the jobs listed on our mobile application. This action will automatically add our student to the list of candidates for that particular job. A human resources admin will select the right candidate given the job's list of candidates.

Furthermore, a student should be able to upload his resume to his profile and win points in the following scenarios:

- when our student applies for a job in our system
- when our student gets selected for an interview
- when our student starts a conversation with a human resources admin
- when our student receives an internship
- when our student receives a full-time or part-time job opportunity

A cubic growth within a given context will define the growth rate of our scoring system, such as when a student applies for a job, he/she will receive 50 points from our system, yet if a student gets a full-time position, he/she will earn 200 points.

The system will provide functionality to manage a given job or a given user profile. It should also provide information about the system itself such as version and policies.

2.3 User Classes and Characteristics

Our application provides two types of users: students and human resources administrators. The term "user" describes both types and allows us to describe common features. Each type of user has unique features associated.

A student is able to apply to given jobs, while a human resources admin manages a given job and selects a student from the list of candidates. Both roles are able to view jobs and users from our system. Only an human resources administrator can read an applicant's resume.

2.4 Operating Environment

The platform supported by our development team is iOS, any mobile application deployed by our organisation will run only on an iOS operating system on Apple's products. Our server will run in a UNIX environment, in isolation.

2.5 Design and Implementation Constraints

A valid and stable internet connection represents a primary constraint in our system. A user of our mobile application will not be able to access the data stored on our external server without an internet connection. Any fetching of data will require an internet connection.

The internet connection's capacity limits our system's performance, since fetching data requires an internet connection. Our local data caching mechanism is limited; the caching will take place ones per session.

2.6 Assumptions and Dependencies

Our primary assumption is that our mobile product application will be used only on currently supported Apple devices that have enough performance.

Another hypothesis is that our user will provide us with a valid email address, to notify him/her of different actions provided by our human resources administrators.

We depend on Apple's application verification system, which provides us with needed permissions to publish our mobile application on the App Store. Furthermore, we depend on Spring, the company which provides us with the necessary framework to create our server.

Chapter 3

External Interface Requirements

This section provides an overview of our user interface and describes the inputs and outputs our system accepts within a given context.

3.1 Hardware Interfaces

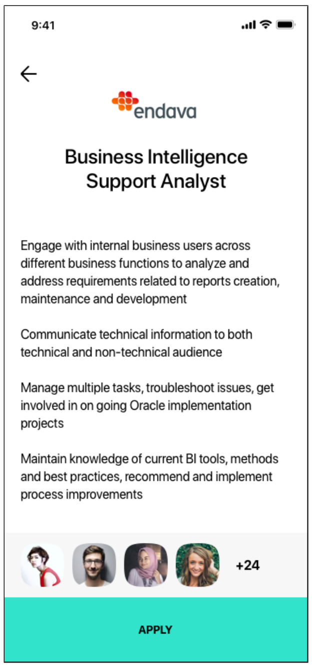
Our external service does not require a specific hardware interface, since we use Kotlin, a portable high-level programming language, as our primary technology. The mobile application requires an iOS device as a hardware technology, in our production environment.

3.2 Software Interfaces

Our mobile application communicates with our external server using a http communication protocol; a given user can read, create, update or delete entities from our database using this communication protocol.

Our database connects with our server using Spring; any repository from our system provides a service to be consumed by our local mobile application. Each entity in our product has his service with allows a given user to perform on a given data entity.

3.3 User Interfaces



9:41

endava

Business Intelligence Support Analyst

Engage with internal business users across different business functions to analyze and address requirements related to reports creation, maintenance and development

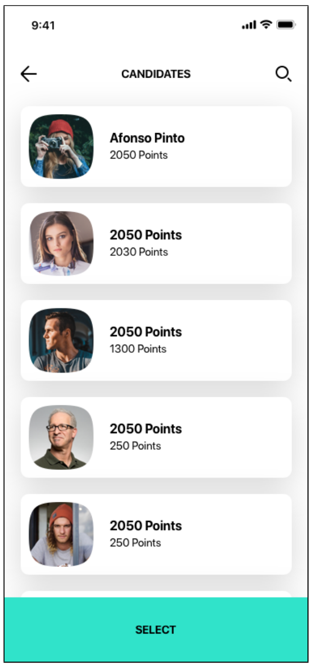
Communicate technical information to both technical and non-technical audience

Manage multiple tasks, troubleshoot issues, get involved in on going Oracle implementation projects

Maintain knowledge of current BI tools, methods and best practices, recommend and implement process improvements

+24

APPLY



9:41

CANDIDATES

Afonso Pinto
2050 Points

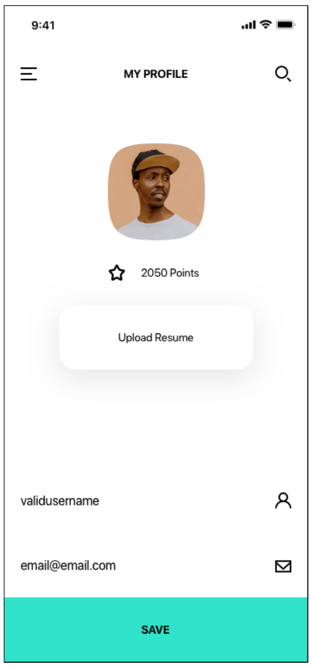
2050 Points
2030 Points

2050 Points
1300 Points

2050 Points
250 Points

2050 Points
250 Points

SELECT



9:41

MY PROFILE

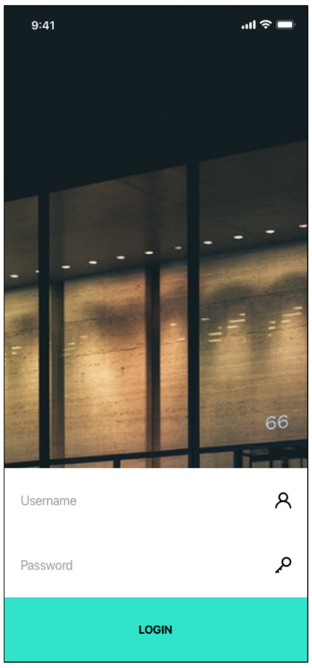
2050 Points

Upload Resume

validusername

email@email.com

SAVE

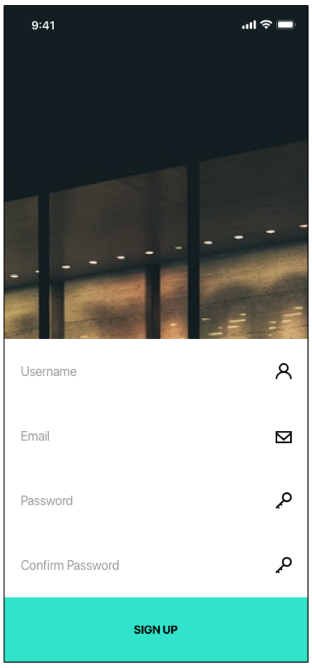


9:41

Username

Password

LOGIN



9:41

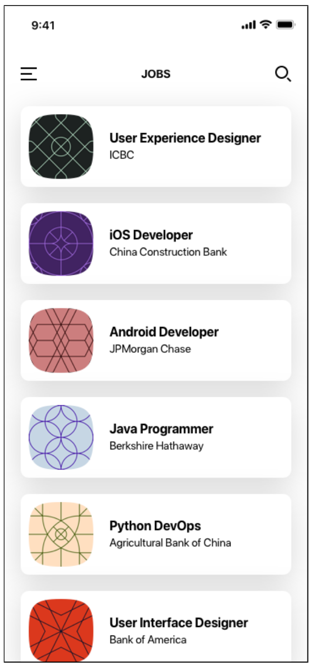
Username

Email

Password

Confirm Password

SIGN UP



9:41

JOBS

User Experience Designer
ICBC

iOS Developer
China Construction Bank

Android Developer
JPMorgan Chase

Java Programmer
Berkshire Hathaway

Python DevOps
Agricultural Bank of China

User Interface Designer
Bank of America

Three categories divide our user interface: register, job and user. The "register" category presents our registration and identification methods. The job category presents a screen that will enable a user to view a given job, a screen with a list of all our candidates for a given position and a method which will list the available jobs from our system.

Furthermore, our final category presents our user profile and provides information about our mobile user. This screen allows our user to update his/her email address or username from our database.

Chapter 4

System Features

4.1 Job Listing Feature

4.1.1 Published Jobs Listing Feature

Description 4.1.1.1. *A given user must be able to view all the jobs published on our system. For each job posted, its information must be available for any given user.*

Condition 4.1.1.1. *The given user has to navigate to "Jobs" screen to view the list of jobs published on our system.*

Rationale 4.1.1.1. *The job entity is an essential component of our service; it provides the value of our product. To allow students to apply to a given job, we must provide them with the appropriate data. Furthermore, this feature allows the existence of competition between students and companies.*

Requirement 4.1.1.1. *Any given job published on our platform must be visible to all platform's users.*

Requirement 4.1.1.2. *Our system must fetch maximum 50 jobs per page to improve performance when fetching data.*

Requirement 4.1.1.3. *A given user must be able to fetch any page of jobs from our external server.*

4.1.2 Jobs Ownership Feature

Description 4.1.2.1. *A given human resources administrator must be able to view the jobs he published on our platform*

Condition 4.1.2.1. *The given human resources administrator has to navigate to "My Jobs" screen to view the jobs he published on our platform.*

Rationale 4.1.2.1. *A human resources administrator may want to update or delete a published job. If the administrator has to select a student for a given job, we must facilitate this process by applying this ownership filter on the list of jobs from our service.*

Requirement 4.1.2.1. *Our system must be able to filter jobs by a publisher.*

4.1.3 Search Jobs Feature

Description 4.1.3.1. *A given user should be able to search for a job in our system by the job's company or position.*

Condition 4.1.3.1. *The user should type the search term in our search bar on our "Jobs" screen.*

Rationale 4.1.3.1. *Only particular job positions interest our students. Some users desire to apply for jobs at a particular company.*

Requirement 4.1.3.1. *Our service must be able to filter jobs by the publisher's company.*

Requirement 4.1.3.2. *Our service must be able to filter jobs by their position such as "Developer", "Designer" and others.*

4.2 User Profile Feature

4.2.1 Profile Resume Feature

Description 4.2.1.1. *A given student should be able to upload his resume and attach it to his user profile.*

Condition 4.2.1.1. *The student should navigate to "My Profile" screen and type on "Upload resume" button.*

Rationale 4.2.1.1. *A human resources administrator may be interested in the abilities of a particular student, to view them he needs a resume. Some companies require experience, special abilities or other criteria when hiring students for their internship positions.*

Requirement 4.2.1.1. *Our service must provide a method to upload files to our server.*

Requirement 4.2.1.2. *Our system should allow only PDF documents to be uploaded to our server.*

Requirement 4.2.1.3. *At a given time $t \in Time$ only a given PDF document per student profile should be allowed to exist on our external server.*

4.2.2 User Points Feature

Description 4.2.2.1. *A given student should be able to view the amount of point he/she earned on our platform.*

Condition 4.2.2.1. *The student should navigate to "My Profile" screen.*

Rationale 4.2.2.1. *A given student should be aware of our gamification system and may want to track his progress while looking for an internship.*

Requirement 4.2.2.1. *Our service must provide the score of a given student profile.*

4.3 Job Posting Feature

Description 4.3.0.1. *A company should be able to create and publish a job opportunity our platform.*

Condition 4.3.0.1. *A human resources administrator should navigate to "Jobs" screen and tap on "Create job" button. After filling in the given form, the administrator should tap on "Publish" to make his job draft visible to all our users.*

Rationale 4.3.0.1. *Since jobs entities provide value for our users (most of them students), we need to create a mechanism which supplies those types of entities.*

Requirement 4.3.0.1. *Our service must provide a form to allow human resources administrators to create new jobs and publish them on our system*

4.4 Candidates Listing Feature

4.4.1 Job Candidates Listing Feature

Description 4.4.1.1. *Given a particular job, a list of all candidates should be available for the job's administrator to review and select candidates for interviews.*

Condition 4.4.1.1. *A human resources administrator should navigate to one of the jobs published by him and scroll until the list of candidates is visible on the screen.*

Rationale 4.4.1.1. *An administrator needs to select a few students who applied for a given job, to schedule an interview. The list of applicants will help our user find the right students for a given internship.*

Requirement 4.4.1.1. *Our service must provide all the student who applied on a given job.*

Requirement 4.4.1.2. *Our system must fetch a maximum of 50 students per page to improve performance.*

4.4.2 Sort Candidates By Score Feature

Description 4.4.2.1. *Given a list of candidates for a particular job, our system must be able to sort the candidates based on their score, provided by our gamification service.*

Condition 4.4.2.1. *At least one student has to apply for the given job to sort the list of job's candidates.*

Rationale 4.4.2.1. *An administrator may be interested in relevant candidates, our scoring system will indicate top performers and will allow our administrator to filter the list of candidates efficiently.*

Requirement 4.4.2.1. *Our service must sort the list of candidates based on their score.*

Requirement 4.4.2.2. *If the score of two candidates is equal, the next sorting criteria will be based on the date on which they applied at the given job.*

4.5 Notification Feature

```
Feature: Selection Notification

    @Notification
    Scenario: Notify Candidate If Selected
        Given a job's candidate
        When a human resources administrator selects the candidate
        Then the candidate receives an email

    @Notification
    Scenario: Notify Student If His Score Changes
        Given a student
        When student's score changes
        Then the student receives an email
```

Figure 4.1: Notification Feature

Rationale 4.5.0.1. *Any action which influences a student's score should be considered important since it affects the student's chances to get an internship. Ones selected for an interview, the human resources administrator, needs to contact the given student to schedule the meeting.*

4.6 Confirmation Feature

Rationale 4.6.0.1. *The confirmation system helps us to filter invalid user accounts and maintain the integrity of our database. It decreases the number of fake user accounts, keeping our data relevant for our costumers.*

```
Feature: Email Confirmation

    @Notification
    Scenario: Confirm Email After Registration
        Given a new user
        When user registers on our service
        Then send confirmation email to user's email address

    @Security
    Scenario: Account Confirmation
        Given a confirmation email
        When user taps on confirm button
        Then user is able to apply for jobs

    @Security
    Scenario: Account Not Validated
        Given user does not confirm his account
        And 30 days passes since his registration
        Then user account is deleted from our system
```

Figure 4.2: Notification Feature

4.7 Scoring Feature

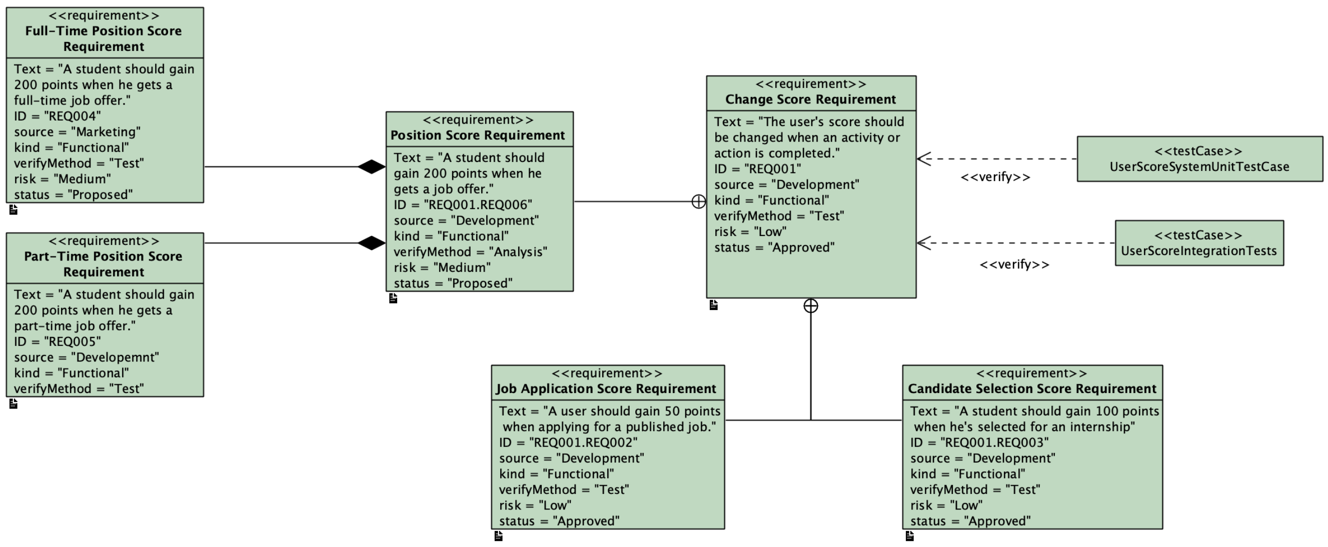


Figure 4.3: Scoring Requirements

Rationale 4.7.0.1. *The scoring system allows us to filter and sort a job's candidates; this method provides relevant students to a human resources administrator who helps our service grow. Besides, the scoring system provides a gamification effect over our service which motivates students to take different actions such as applying for a job opportunity.*

Chapter 5

Other Nonfunctional Requirements

5.1 Performance Requirements

Requirement 5.1.0.1. *Any list of items should load on the user's screen within 2 seconds, depending on the internet connection.*

5.2 Security Requirements

5.2.1 JWT Authentication Feature

Description 5.2.1.1. *A user should be able to access our platform only with a unique JSON web token, generated when he logs in our system.*

Condition 5.2.1.1. *Our registration or authentication system should provide a JSON web token to validate a user's session within our service. A user has to login or register to get the required token.*

Rationale 5.2.1.1. *Authentication tokens provide us with flexibility and security; our service will be safer to use for our end user, and at the same time other 3rd party services will be able to access a user's session.*

Requirement 5.2.1.1. *Our service must provide a JSON token to any authenticated user.*

Bibliography

[1] Elliott, Sr., R. A. and Allen, E. B. (2013). A methodology for creating an ieee standard 830-1998 software requirements specification document. *J. Comput. Sci. Coll.*, 29(2):123–131.