Requirements Analysis Document

UWA Interview Guide Builder Project

Preface:

This document addresses the requirements of the interview guide builder project proposed by the UWA Talent Acquisition Team. The intended audience for this document are the developers and the clients of the project.

Target Audience:

Client, Developers, Auditor

CITS3200_6 Members:

- Caitlyn Chan - 22087862
- Timothy Eitel - 22091141
- Andrew Ha - 22246801
- Hao Ho - 22250127
- Farruh Mavlonov - 22252282
- Zhen You - 22036933

MILESTONES

- **Sprint 1 Deliverables**
  - Scope of work
  - Skills and resources audit
  - Project acceptance tests
  - Risk register
  - Stories for sprint 2 and intermediate acceptance criteria
- **Sprint 2 Deliverables**
  - Plan for database schema
  - Populate local database with competencies, key actions and interview questions
  - Front-end wireframes/design for UI/UX
  - Back-end server setup
  - Tests implemented for code
- **Sprint 3 Deliverables**
  - Web application deployed
  - Source code for web application
  - User/system documentation
  - Authentication/login

1.0 General Goals

The objective of our proposed web application will be to minimise the amount of time taken to compile the interview guides. It will automate and simplify the process by hosting a digital database with the competencies, interview questions and key objectives.

These competencies will be organised alphabetically on a searchable index however, there will be additional mechanisms in place that will provide the talent acquisition team with the ability to search and filter the results. The selected competencies will then be automatically merged into two word documents that will be utilised as an interview guide and a selection matrix by an interview panel.

2.0 Current System

The current system that the UWA Talent Acquisition Team have in place involves a database of competencies, key actions and relevant interview questions that are stored in the form of multiple word documents. The team need to manually assemble the interview guide templates by tediously copying and pasting the important information onto another separate word document.

The consolidation and manual formatting of all the information into a single user-friendly interview guide and selection report for the interview panel is subsequently tedious process.

Due to the high volume of interviews that need to be conducted by the interview team every week, this manual administrative process is quite taxing. This is exacerbated by the inconsistent templates, formatting and processes of the aforementioned interview guides and selection reports across the team.

3.0 Proposed System

3.1 Overview

Our proposed digital solution will be a web application that will be hosted online. The web application will be deployed on Heroku (a cloud application platform) once the web application works on localhost. As the web application will be deployed, a remote database will be utilised (mLab an online version of mongoDB) whilst mongoDB will be utilised for the local database.

The back-end (server side) will utilise the Express.js framework. Express is a minimal and flexible Node.js web application framework that allows us to create a web server that is more maintainable and flexible.

The front-end (client side) will utilise the Vue.js framework and Material Design using the component framework, Vuetify. Vue is a progressive front-end framework for building complex user interfaces. Material Design is a design language with an adaptable system of guidelines, components and tools that will aid with the design of the user interface. Vuetify is a component framework that is easy to use, is well supported on many browsers and it allows developers to focus on the core functionalities of the web application whilst also being readily customisable.

In regards to the colour scheme and branding of the site, it will adhere to the standard UWA colour coding palette and branding style provisioned to us by the UWA Talent Acquisition Team.

3.2 Functional Requirements
The objectives of the functions of our subsystem will fulfill the following functional requirements:

1. Host a database of competencies and questions (approximately 20 competencies, 5 questions per competency)

2. Provision of a tool for selected competencies and questions to merge into two separate Word documents:
    - An interview guide
    - A selection matrix

The aforementioned functionalities will accommodate a small number of users (5 to 6 human resources team members) that will utilise the system on a regular basis. The generated documents would then be distributed to relevant users, namely the members of the interview panels.

3.2.1 Back-end / Database Requirements
The database will need to provide users with the ability to add, remove and edit competencies and their respective key actions and interview questions. Scalability and ease of maintainability of the database is subsequently very important when it comes to constructing the database.

The database will not only need to store the competencies, key actions and interview questions but it will also need to have a database of parameters and metadata tags that will enable users to filter their search results.

Due to the confidential nature of the information being stored on the database, ensuring that the database and web application is secure will need to be taken into consideration when constructing the database and server side. We will be utilising best practices from Microsofts' security development lifecycle (SDL).

3.2.2 Front-end Requirements
Regarding the front-end, the user interface will need to allow users to be able to select multiple competencies which will then query the database for the relevant key actions and competencies that will then be merged into a standard template. This template will be used to formulate the interview guide.

The front-end will need to have an index that lists all of the competencies and allow users to search for the relevant competencies. It will also need to provide users with the ability to select tags and parameters in order to filter the results.

The overall UI/UX design will be simplistic and intuitive so that new users won't need to account for a steep learning curve in order to utilise the web application.

3.3 Nonfunctional Requirements
For this section, list out the non-functional requirements of your subsystems in the following headings

3.3.1 User Interface and Human Factors

The system will be in use by a small amount (5 or 6) of team members from the talent acquisition team, of which all users are of a novice level with computer systems. Some initial training will be required to teach the users how to select competencies and where documents (interview guide and selection matrix) are generated. However the front end of the system should be intuitive such that the system can be used by anyone with no training.

Errors concerning the database are the most dangerous in this system as deleting or corrupting the database means losing new data that was not committed paper, so there will be multiple protections put in place to prevent this. A lesser concern is errors involving the word document generations as these can be fixed manually in post.

3.3.2 Documentation

From the client's perspective, they will require documents involving weekly progress (GANTT chart), changes to the scope and stories and final state of the system.

The designers will require documents involving the model of the entire system, testing implemented and future additions that can be added after the deadline. These documents should also be supplied to the client in the case they want to get other teams to work on it. The only document that will be required for users is a manual on how to use the system.

3.3.3 Hardware Consideration

Since this is a web application it will work on any hardware that supports browsers.

3.3.4 Performance Characteristics

There are no speed, throughput, or response time constraints on the system however it would be preferable for documents to be generated as fast as possible. There are no size constraints since the data that is being stored (40-50 competencies) and data being produced (2 word documents) are small in size.

3.3.5 Error Handling and Extreme Conditions

Testing will be conducted internally using Cypress (a testing framework).

3.3.6 System Interfacing

The only input required is from the keyboard and mouse. The only output that will be generated are the two word documents which will be emailed to a nominated user.

3.3.7 Quality Issues

We will be implementing unit testing, integration testing and end to end testing using Cypress.

3.3.8 System Modifications

Most of the current infrastructure will become obsolete as the list of competencies, questions and key actions will be stored in a database instead of a word document. Also the current system of manually creating will become automated by the proposed system.
The most likely part of the system that will be improved in the future after the project is the output. Eventually the system can be upgraded in such a way that word documents will not be required and interview guides can be completed on an electronic device.

3.3.9 Physical Environment

The web application will be hosted online and will be accessible on all the following web browsers: Chromium (Chrome, Edge Insider) , Firefox , Edge and Safari 10+.

3.3.10 Security Issues

Since the data in the database is to be used only by panel members and the user team, there will need to be a verification system to only allow certain people into the frontend to access the database.

3.3.11 Resource Issues

The data will need to be backed up at the end of every session when anything is added to the database. This back-up would take the form of a word document that's generated with all competencies, questions and key actions. The responsibility of backing up will be on the user to press a button on the web app for the back up. In regards of deploying the system to heroku, the responsible party will be on the developers. If there are any maintenance requirements of the system, the client will contact us.

3.4 Constraints

The main constraint on this project is time. With the project length being short, program languages will mostly be decided on how fast and easy it is to learn and implement. This constraint also affects the number of user functionalities that can be completed.
Since this product will be for the University of Western Australia, the UI will need to be consistent with UWA standard.

3.5 Scenarios

Compulsory

1. As a user, I want a database to store competencies, questions and key actions. So that there is a digital copy and to be accessed to create the interview guide.
2. As a user, I want to be able to select competencies and questions, to generate the interview guide.
3. As a user, I want to be able to input (type) the job title, to generate the interview guide.
4. As a user, I want to add and delete competencies, questions and key actions, so that I can grow the database in the future.
5. As a user, I want to input (type) candidate's names, job title, panel members and interview date, to generate the selection matrix.
6. As an IT support, I want to be able to run tests on the systems, so when I'm contacted about a problem I can identify what it is.

Extensions

1. As a user, I want to filter the questions in terms of relevance to the level of job, to streamline the searching of the database
2. As a user, I want to be able to do search for competencies, to streamline searching of the database
3. As a client, I want to be able to generate a word document of competencies, questions and key actions, so that there is a backed up copy locally.