System Requirements Specifications

JobSeeker

McMaster University

Software Engineering Department

2XB3 Software Engineering Practice and Experience

Lab 03 Group 03

Zihao Du duz12@mcmaster.ca

Gengyun Wang wangg40@mcmaster.ca

Senni Tan tans28@mcmaster.ca

Wenzhi Wang wangw95@mcmaster.ca

Table of contents

The document explains what is the application domain, the goal of developing the design and who are the stakeholders. And functional and nonfunctional requirements the design shall meet are listed here to specify the behaviour of the design. The document also has requirements on the development and maintenance process like priorities of required functions.

Assumptions

We assume the users' age are older than 18 years old, because people usually start to apply for a full time job at such age. There are no constraints about users' nationalities or addresses, because the data that our data set collected from are worldwide.

Domain

This software will provide following services for several user types:

- **Students and Unemployed people**: This group of users usually are eager to find jobs with high potential as a good starting point of their career path. Therefore, after users select the job-field they want to search and their addresses, our software will give a list of satisfactory jobs from the adjacent areas, sorting by the job-prospect. Users can also type the NOC Code of the job they are interested to see the details.
- **People who want to change their job**: This group of users usually want to get a new job that relates to their previous working experience. Such users can find the relative works after they type the field and region they want to search.
- **Job market researcher:** This group of users usually want to analyze the performance or trend about a specific field or a specific job in a certain region and certain year. This goal is similar to the goal of the first group, but the researcher also wants to know past data, not only the latest data. Therefore we will set a year-input, the user can search the data in the period they type.

Functional requirements

The design shall take the category of job and city the user is interested in as input, and show a list of jobs in the city, or the nearest region if the city is not in the dataset. The User can check detailed information like outlooking and number of relevant jobs in the same region of a specific job.

I/O requirements

- The design shall be able to take CSV files with over one million rows as input without overflow.
- The design shall allow the clients to search as many times as they want without getting any errors.

Searching requirements

- The design shall be able to find out jobs of a certain category in a certain region.

Sorting requirements

- The design shall have a changeable compareTo method to tackle different two of sorting problems: sorting alphabetically and sorting with respect to integer.

Map requirements

- The design shall be able to determine the nearest region of the user's input location in the dataset.

User Interface

- The design shall have a user interface that takes the user's inputs and stores them to control other modules and visualize the outputs.

Nonfunctional Requirements

- Reliability --- The product should work with the satisfaction of design specification most of the time; the product will be tested by each group member and 99% of the test cases must be passed.
- Robustness --- The product would handle the unexpected inputs from the user; if the user gives the unexpected input, the input should not be taken and an error message will be displayed.
- Performance --- The program should run as fast as possible and the outputs should look generated; the algorithm used should have a running time of at most nlogn, and the outputs should be displayed in a sorted alphabet order or by the order that the user selected in the filter.
- Usability --- The product should be easy to use for the user and there must be proper instructions that guides the user to use the product; the prototype will be given to volunteers and the volunteers are invited to use the product, then the volunteers will be asked the question "do you think it is easy to use?", at least 95% of the volunteers should say yes.
- Maintainability --- The product should be implemented in a way that makes the product modified after its initial release easy; after the initial release, the group members will improve the implementation by the feedback and if a total four of group members feel that the product is easy to be modified, then the product is maintainability.
- Portability --- The product should satisfy that it can run in different environments, such as the operating system; after the first prototype , if the product can work on the system of Windows, Linux and Mac OS, then the product is portability.
- Understandability --- the requirements, design, implementation, documentation of software product, etc. should be easily understood; find volunteers in the software engineering department and invite volunteers to read the design implementation,

documentation, etc; if over 90% of the volunteers think those are easy to understand, then the product is understandable.

- Accuracy of results --- the results for each input should be accurate; every output should be the relative information of the job that the user inputs, no information mismatch happened.
- Physical constraints
    - barrier: the user could only enter the string input that follows the instruction; if a user enters the string that does not match the instruction, the string will not be taken as an input and an error message will be displayed.
- Operating constraints
    - If the user gives the input of category, the user can only select one of the categories on the list given.
    - The user must have a computer device such as computers and notebook computers, not smartphones or tablet or any other devices.
    - Human-computer interface issues --- the result must be displayed on a screen; therefore the user must have a computer with a screen.
    - The user must have a computer with one of the following systems: Windows 7, 8, 10, Linux, Mac OS.
    - The user must use a computer which is connected to WiFi or local network when using the product.

Requirements on the development and maintenance process

Quality control procedures

- Multiple test cases would be chosen randomly to ensure each function works properly.
- Marginal input cases including non-existent job titles, invalid addresses, etc, should also be considered.

- During the testing process, each module will be tested by both the programmer and the tester.

Priorities of the required functions

- The searching, sorting, and graph modules are the core of the product and should be tested independently by each corresponding programmer. The input and output modules are of less priority in terms of the development process.

Likely changes to the system maintenance procedures

- In order to maintain the system, the team needs to keep its core functions correct and reliable, which involves keeping the data of job titles updated so that the client would not receive outdated or misleading information. The database, however, might not be updated as frequently as our system requires, and the maintenance could include checking the validity of the data set from other sources.
- If the user base expands in the future, it would be important to optimize the algorithms. It is possible that after long term usage, the team would be able to find more common user input cases and the resulting graph from such cases could be stored to improve the efficiency of our system.

Other requirements

- During the development process, a certain order based on the dependence of each module should be followed to increase productivity.
- For better quality control results, team members will review each member's code and avoid confusion.

Reference List

Robert Sedgewick., & Kevin Wayne. (2011). *Algorithms*(4th ed.). Boston, MA: Pearson.

"IEEE Recommended Practice for Software Requirements Specifications." 1998. PDF file. http://www.cse.msu.edu/~cse870/IEEEXplore-SRS-template.pdf