**Chapter 3**

**Requirements Specification**

Requirements specification is a process of how the team gain proper and reasonable

requirements from stakeholders. It is the foundation of this project [8]. Requirements

should be precise and complete for team to develop further steps based on them. This

chapter consists of use requirements, system specification and user story. After analyzing and collating data gained from last stage, specific requirements and system specification have been confirmed. Section 3.1 and 3.2 will include this. Then, in Section 3.3, user story is produced to further enhance the team’s understanding of requirements.

**3.1 User Requirements**

Functional requirement describes what a stakeholder needs to be able to do. Non

functional requirement constraints on what a stakeholder needs to be able to do [21].

**Functional Requirements**

1. Users want to learn sorting algorithms through the software.

2. Users want to watch animations of the process of sorting algorithms.

3. Users want to understand and prove the correctness of sorting algorithms.

4. Users want the software to support multiple languages

**Non-functional Requirements**

The software can animate bubble sort, selection sort, insertion sort, merge sort, quick sort and heap sort.

**3.2 System Specifications**

Functional specification is what the software must do to meet the user requirements.

Non-functional specification constraints on what the software must do to meet the user

requirements [21].

**Functional Specification**

The list is divided into two parts: core features and optional features. Core features are

essential to the operation of the software and optional features are additional functionality.

**• Core Features**

1. Hierarchical learning feature

A user shall be able to choose a difficulty level according to individual mastery

level of algorithms.

1. Sorting animation related features

* A user shall be able to select a sorting algorithm to learn.
* A user shall be able to view the animation, which shows the sorting process.
* A user shall be able to view explanations of each step of sorting algorithms.
* A user shall be able to start from different time frames of an animation.
* A user shall be able to customize the array of numbers to be animated
* A user shall be able to ask the software to randomly generate input.

1. Correctness related features

* A user shall be able to test an algorithm through several sets of legal inputs

to see if it can work correctly.

* A user shall be able to see explanations of sorting algorithms’ correctness.
* A user shall be able to choose from several sets of illegal inputs.
* A user shall be able to test whether an algorithm can work out its job or not.
* A user shall be able to view proof explanations of a sorting algorithm.

4. Open source

* A user shall be able to access the source code of this software.

**• Additional Features**

1. A user shall be able to view a user guide of software operations.
2. A user shall be able to take a tutorial of basic algorithms.
3. A user shall be able to view the pseudo-code of sorting algorithms.
4. A user shall be able to view the programming code of sorting algorithms.
5. A user shall be able to compare different sorting algorithms' time complexity.
6. A user shall be able to view history of the latest learned sorting algorithms.
7. A user shall be able to view individual learning progress.
8. A user shall be able to download learning notes provided by the software.
9. A user shall be able to do after-class exercises.
10. A user shall be able to share the information of the software.
11. A user shall be able to write feedback to developers.

**Non-functional Specifications:**

1. Usability

The software supports English and Chinese.

The software provides programming code in C, Python, and Java language.

The software supports Windows and Mac Operating Systems.

The hardware requirements for basic performance is 200MB storage.