CSC 490 – Spring 2022

Requirements Document (Delivery No. 1)

Due Date: 6 / 3 / 2022 (soft copy uploaded to Blackboard)

During the first phase of your "CSC490 Software Engineering" project, you will work on creating the specifications for a new software entitled "**Virtual Learning Assistant**". The main description (problem definition) of the desired software is described as follows:

*Several students at LAU requested a tool that will assist them in their studies. Among the main issues that many students (especially freshmen) are facing is that they don't know if they are studying enough for each course. In some courses, the instructor could specify the required study hours for a certain **Chapter/Section**. The tool would then track the student's study hours on the LMS and compare the students' **total** study hours for the Chapter/Section with the required number of hours specified by the instructor. In other cases, the tool would collect and analyze the study hours of each student in the course, calculate the **average**, and inform each student if her/his study hours are well below the average. The second main thing that the tool would focus on is to make sure that the student is studying/performing each lesson/task **on time**. Many students get lost when there are a large number of tasks to do, and may forget to do or complete some tasks. Hence the Virtual Learning Assistant (VLA) would track the **status** and **completeness percentage** of each **task/lesson** and notify the student of the status of **incomplete/remaining** tasks (the student should be able to configure the **times/conditions** on which she/he would like to receive a notification). Another major issue that most students stated is that they spend a lot of time searching for a specific **topic or title** within the course **content and materials**. Several students expressed their need for a help tool that can let them find the specific **pages or slides** that contain the topic/keyword they are searching for.*

For your CSC490 course project, you are required to design and implement the described VLA software. In this phase of the project, we will focus on the software specification, i.e., the requirements document. You are required to work in groups of three students for this project. Your project should satisfy the following main conditions:

1. The VLA should have at least **five** main objectives. You are given three objectives in the problem definition stated above. You are required to add at least two more main objectives (for projects that contain four students, you should have at least **six** main objectives).
2. In order to find the additional objectives, you need to conduct project team meetings, discuss the most critical studying issues that the VLA can assist you with, and decide which issues can be implemented by the VLA. You should also conduct interviews with students at different levels (freshman, junior, senior, graduate) and ask them what are the most

important studying issues they need help with. Conducting interviews with students at different levels will enrich your requirements' collecting experience and make your software helpful to a wide range of university students. Your overall software importance will largely depend on the objectives that you will specify at this stage, so select carefully.

3. If needed, you can replace **one** of the three main objectives that were stated in the problem definition above with another objective that you select. You can do this if you feel that you discovered new objectives that are more important than the three defined objectives.
4. In addition to the objectives, you need to identify the conditions and constraints that will be applied in your software. You can refer to Section 1.4 "Operational constraints" in the *Mentcare Requirements Document* that was provided by the instructor (under Blackboard → Slides → Chapter 4 → Documents → Mentcare Requirements) to obtain an idea of how conditions and constraints are expressed. Constraints can be related to the rules/regulations that the software must comply with, the other systems that your software should interact with, system performance, user-related constraints, code reuse, time constraints, hardware-related constraints, parallel operations, programming language dependency, reliability, security, privacy, etc.
5. You will be following the **Waterfall Model**: your software specification must be well-revised to ensure that it is complete and final, as you will not be able to modify it in the next phases.

After you specify the main objectives and constraints of the VLA, you need to write the "User Requirements" of your system. You have to express the user requirements in a clear and standard format. The user requirements should be deduced from the system objectives and constraints. For each objective/constraint, you identify the specific system functionality (or functionalities) that is (are) required to satisfy the objective/constraint, and you express each functionality in a separate user requirement. Below are some guidelines:

1. Do not mix requirements, make sure every requirement is discussing **ONE** functionality (of a certain objective/constraint).
2. Indicate the source of the requirement (i.e., the source from which you deduced the requirement): discussion, interview, Internet, etc.
3. Indicate the type of the requirement: functional or non-functional. In addition, if the requirement is a domain requirement, add a symbol that indicates that.
   An example of a domain requirement could be: The VLA will use the notification feature in Moodle to send notifications to the student.
4. Use proper technical English: ensure that the language is clear and simple so it can be read by customers.

In the next phase, you need to identify and categorize the **System Requirements**, which are the expansions of the user requirements. Refer to Figure 4.1 in the Textbook as an example of how system requirements are deduced from user requirements. In addition to expanding the functional user requirements, you need to expand the non-functional user requirements **when needed**.

Hence, if a certain condition/constraint includes a lot of details, you should ensure that you added all these details in the generated system requirements (for example, if you are going to implement your VLA as part of a specific LMS, you need to include all the constraints that are imposed by this LMS). In cases where a set of system requirements belong to a single category (for example, user privacy requirements), you should group these requirements into a separate sub-section under the name of the category. You can refer to Sections 2.1, 2.2, 2.3, 5, 6.1, and 6.2 in the *Mentcare Requirements Document* that was provided by the instructor as examples of such categories.

If a requirement contains multiple cases (if…then); such as Requirement number 3 in the *Insulin Pump Requirements* document that was provided by the instructor; you can express the multiple cases as sub-requirements under the main requirement (see Requirement number 3 in the *Insulin Pump Requirements* document as an example).

The system requirements of the VLA tool should focus on the following aspects (Note that this list is not comprehensive. You might add to it other aspects based on the specific objectives of your project):

- Students' interactions with the VLA (student-related requirements)
- Instructors' interactions with the VLA (instructor-related requirements)
- administrators' interactions with the VLA (administrator-related requirements)
- VLA operations that are executed in the background (i.e., without the student's interaction)
- VLA configurations and options
- Handling of various events that are related to the objectives (processing, data collection, data exchange, etc.)
- Ensuring that the VLA is compliant and consistent with its environment (LMS, other tools it uses, etc.)
- Ensuring the security of the system's data and operations
- Ensuring the privacy of instructors and students (for private data)
- Ensuring that the VLA can be extended and new features can be added to it later

After completing the user and system requirements and categorizing them, you need to create the requirements' dependency graph, in which you connect each requirement to the requirements that it depends on or is affected by. For example, if you connect Requirement A to Requirement B (with an arrow towards B), meaning that if B changes, then A should be analyzed for possible change.

The final step in the software specification is to gather the information that you generated above in the **Requirements Document**. Your Requirements Document should follow the IEEE Requirements Standard that was uploaded by the instructor to Blackboard. Attached to this document is a template that you can use to write your Requirements Document. You can replace the text in the

template (Section title and body) as you require. You can use your own template if you prefer. Overall, your Requirements Document should contain the following **main sections**:

1. Introduction: One or two pages that summarize the overall project, the type of software you are creating, its importance to its users, and the motivation behind it. In this section you write the complete problem definition of the desired software. You can use the problem definition that was given to you at the beginning of this document, but you need to expand it by adding the full details of the system description and the full details of the objectives that you added.

2. Glossary: Definition of the technical terms used in the document.

3. Background: The results of your research on similar software on the Internet, and how the software that you are creating differs from those that have been implemented before.

4. User Requirements: An introduction to the user requirements followed by their details: separate the functional requirements alone and then state the individual non-functional requirements. If a set of user requirements (could contain both functional and non-functional) belong to a certain category, you can include them in a separate sub-section.

5. System Requirements: Includes all the details related to the system requirements. For each requirement, you need to specify its type (functional or non-functional). As stated before, group the requirements that are related into a separate sub-section. You can include a subsection that contains the general non-functional requirements (i.e., those that don't belong to a specific category), followed by a subsection for each category.

6. Requirements Evolution: present and discuss the requirements dependency graph.

7. Conclusion: a short paragraph summarizing what the Requirements Document contains and its importance to the project's success.

8. Appendix (if needed): provide any additional information that elaborates or explains certain parts from the previous sections.