Homework 7: Validation/Customer Review

(Team 3, reviewing Team 2)

1. **Correctness**

In the specific requirements from pages 6 to 14 on the SRS, every requirement represents something that is required to be built for the system. In the essential requirements under the specific requirements, both functional and non-functional requirements features matched with the use cases features in the homework 4 use cases descriptions section found on pages 44 to 67. It seems all use cases found on the Use Case Diagram are listed on the essential requirements, and the use cases descriptions sections. Use cases includes Login, AuthenticateUser, ManageAccount, AccessCampusResource, FindJoinGroup, LeaveReview, FindAddFriend, CommunicateWithUser, VoiceOrVideoCall, EventCheckIn, FindEventAndRSVP, ReportUser, HandleReport, CreateGroup, PublishEvent, TrackGroupPerformance, RewardAttendance. These are all requirements required by the system to be built.

1. **Unambiguity**

Overall, the requirements section of the document was well-articulated and described, and went into much depth to generally avoid ambiguity. However, a few grammar errors in the requirements specification have led to ambiguity. For example, in 3.1.2.4: Interoperability, there was a dangling modifier in the Rationale section. The phrase “By being compatible with Google Calendar, users can mark…” implies that the user is the one who is compatible with Google Calendar, not the software, which is not mentioned in this sentence. In the Hardware Interfaces section of 3.1.3 External Interface Requirements, the description “Anteater Groups will operate on iOS and Android Devices. The prototype will have either one of them based on the developers…” is confusing. The phrase “the prototype will have either one of them based on the developers” is ambiguous as the previous sentence seems disjointed. The phrase “based on the developers” also is vague, as it is not certain what the developers will base something on. Does “based on the developers'' mean that the developers will choose between iOS and Android? The grammatical structure in some areas of the requirements document seemed a bit awkward and led to ambiguity. In sentences such as “This ensures that though regardless of the password being correct, DuoMobile for the specific entry must be approved to proceed further” and “For example, if there is an event happening in 2 days, following the RSVP of the event, users can add it to their Google Calendar”, sections of the sentences could be rearranged. For instance, the first sentence could instead read “..password being correct, the specific entry must be approved via DuoMobile…” As for the second sentence, it is not certain what the goal of including the example “event happening in 2 days” is, as “Users can add an event to their Google Calendar after they RSVP” would be more concise and clear.

1. **Completeness**

Every section of the SRS document: general descriptions - product perspectives, product functions, user characteristics, general constraints, assumptions and dependencies, and apportioning of requirements; specific requirements - functional requirements, non-functional requirements, external interface requirements (user interfaces, hardware interfaces, and software interfaces), and logic data model are all completely addressed in detail in the description and under the appendix section. All diagrams: Goal Model Diagrams, Domain Diagrams (Class Diagrams), and Use Case Diagrams are all included and completed. Goal Model Diagrams addressed the relevant relationships between different activities toward a specific goal. Domain Diagrams (Class Diagrams) illustrate the connection between different entities and actors within the usage model; it also successfully listed each class in the goal models. And finally, Use Cases Diagrams displays the connections between the users and the stakeholders to different use cases features. In conclusion, everything the software system is supposed to accomplish the SRS document has it on there with complete descriptions and use cases.

1. **Verifiability**

The document did not include many measurable information and features, thus, making it more difficult for the system to be verifiable. In describing the amount of users the system should support, the team stated that it should support “as many users as possible.” By leaving out a specific number for this section, section 2.3, it makes it difficult to test the system for this requirement. Although it is left out in section 2.3 , the team did include that it should support “at least 10,000 users” for #42 in their field notes. It would have made this section better in terms of verifying because there is a set number that can be tested against. The general constraints mentioning the various platforms that the system needs to be supported on is great. By mentioning the system needs support on iOS, Android, Google Chrome, Safari, and Firefox, the application can be tested on these platforms to verify that these requirements are met. An addition that would be great to include would be budget and schedule. There seems to be a lack of any mention of a specific number for budget in the document, which makes it difficult to verify what appropriate resources are needed to implement the system. Similar to the number of users supported, the document could include field notes #62 as a non-functional requirement to keep track of the system’s progress and timeline. This document can improve its verifiability by including more measurable goals so that they can test it against.

1. **Consistency**

The requirements of the document are generally consistent with the information portrayed through their analysis models. Group 2 has made clear distinctions between their various functional requirements, indicating the relevant goals, actors, and operations associated with each requirement. Their rationales align with the overarching goals for the system for both functional and non-functional requirements. However, there are instances where sources are not best matched to the specific requirements. For example for FUNC1, and FUNC2, regarding creating groups and joining groups, their sources reference “goal #1 from AnteaterGroups Functionality Outline”, but it seems that these requirements match more with goal model 2, “Create a new group” and “Students find groups to join” goals. However, it is unclear whether “goal #1 from AnteaterGroups Functionality Outline “ refers to their Goal Oriented Analysis Model, as “Functionality Outline” is only mentioned within these requirements’ sources. Furthermore, NONFUNC2 on security, has conflicting dates with their apportioning of requirements section 2.6. NONFUNC2 states that “Users will contact the developers if there are any issues in the first 6 months after launching, and after that contact UCI OIT if there are any issues with login or other technical issues.” Although section 2.6 states “the following requirements may be delayed,” it is not explicit whether “Redirecting support emails to UCI OIT can be done 3 months after the launch of AnteaterGroups” is referencing the same NONFUNC2 requirement. In this case, the time period of when users can contact UCI OIT is both 6 and 3 months after the launch of AnteaterGroups which is inconsistent.

1. **Modifiability**

The formatting of the document is predominantly clear, with sections distinctly labeled and referenceable from the table of contents. The divide between functional and non-functional requirements is distinguishable, and the requirements within each of those categories accurately belong to those respective sections, allowing changes to any type of requirement to only change a specific section of the document, rather than the whole. The user interfaces cover a wide range of system requirements, without overlap of the individual functions. Therefore it would be easy and simple to go in and change the user interface of one interface without having to change another. As for the functional requirements specifically, there is an instance in FUNC3 and FUNC6, in which FUNC3 states “Student users shall be able to view their friends within a friends list, and are able to contact friends directly through text, voice, and video chats” and FUNC6 states “The system shall allow UCI students to contact one another through text, voice, and video calls.” With both of these requirements discussing the ability to contact through text, voice and video chats/calls, if the modes of communication were to change, more than one requirement would have to be found and changed. Mentioning of the communication modes are redundancy, and speaks of the same information in more than one place.

1. **Traceability**

The developers clearly stated their sources for each and every one of their requirements. They specified goals from their outlines, and enumerated which questions were used from the requirements elicitation sessions. Additionally, they not only specified goal models by numbers but also their titles to avoid any confusion. These sources are easily found by looking at the table of contents placed at the top of the requirements document. In the appendix section, each analysis model was labeled by the type of model as well as which homework the model originated from. Not only were the elicitation sessions marked in the table of contents, but they were labeled based on the date, week of the quarter, and class section (discussion or lecture). All of these traits allow for seamless traceability as the dates are clearly listed in chronological order.

1. **Understandability**

The document, overall, is understandable as it includes all the important information for developers, such as: stability, priority, rankings of importance, rationale, annotations, descriptions, dependencies. The priorities and stabilities of each requirement is rated on a low, medium, high scale, so this gives developers a good sense of the importance for each feature.

For each rationale and description, they all accurately describe what the feature should do in a short, clear, and concise manner. For instance, contacting students via “text, voice, and video call” clearly describes the various methods for the functional requirement 3.1.1.6 Communication Modes. The figures are all mostly labeled and correctly describe what they are representing. Section 3.1.3 is a great example of labeling each figure with easy and simple to understand descriptions. By labeling the figures, it helps developers better understand what and how the system should look like. However, figure labeling should be added everywhere where it is applicable. The figures on page 3, 14 and a majority of the appendix would benefit in terms of understandability if they each included a label. A few of the dependencies can be made more specific and elaborated on more. The searching and joining groups functional requirement’s dependency states that “groups must be functional within the system.” There are several similar dependency statements, which can be improved by making them more specific, including more information about the feature, if any condition needs to be satisfied beforehand, or even if it relies on a third party system.

1. **Overall Review Process**

Our overall review process included referring to the lecture slides for each attribute, comparing sections against the rubric for our homework 6 assignment, and checking if all the content was included based on the systems requirement specification template. We went through the entire document for each attribute, one at a time. This made evaluating the document easier to manage and validate since there was only one attribute we were focusing on. On the other hand, the most difficult aspects of reviewing the document would be trying to cover all of the content and deciding whether or not it satisfied each attribute.

To complete the section for correctness, we looked at the specific requirements and essential requirements to check for every requirement that represents something required of the system to be built. To complete the section for completeness, we looked at the overall SRS document and checked for completeness in each of the sections and descriptions of all elements in each section.

To complete the section for unambiguity, we read through the different requirements and focused on the description and rationale for each, as those tended to have the most writing. We focused on the overall flow and our conceptual understanding of the text, and highlighted the sections which seemed confusing. Then, we went back to these highlighted sections and thought about why we brought attention to them. We focused on grammatical errors and the structure of the sentences. This section was not too particularly difficult as there were not too many errors or ambiguities. For the traceability, we looked at the source sections of the requirements and verified them by looking through the appendix. This part was additionally not very difficult, as the document made it clear where we were able to verify the sources.

When we checked the document for consistency, we had to be mindful of the specific details that each requirement entailed. The description, use cases, rationales, sources, and dependencies of each requirement in particular could hold possible instances of conflict or contradiction, as these references may not actually align with the information provided. We had to check if the requirements descriptions were consistent with the descriptions provided in their analysis models. Furthermore, we had to check if the requirements’ dependencies also matched their respective use case preconditions. Sources referring to goals had to be traced down to the goal models and we had to discuss if the rationales aligned with the sources mentioned. The most difficult part of checking for consistency was the need to look at every analysis model to ensure there are no contractions within the individual requirements specifications and the requirements’ details compared to one another.

Looking over the document considering modifiability was also not too difficult. The document was clearly structured to begin with, keeping functional and non-functional requirements distinctly divided. We had to check that each section held information unique and relevant to the section title, as going in to change something pertaining to the document’s information in the future, would require clarity in where to specifically make that change. For the most part, there were few redundancies that we found that would require going into multiple sections to make a single change.