### Inf 43 – Homework 1

##### Student Name: Peter Anteater

#### Student Number: 12345678

|  |  |  |
| --- | --- | --- |
| Awarded Points | Maximum Points | Document Aspect |
|  | 15 | Clarity of writing (spelling, grammar, sentence construction) and Clarity of expression (flow, structure, making logical arguments). Roughly 7.5 each. |
|  | 15 | Introduction / Executive Summary (can be different sections or combined into one) |
|  | 7.5 | Application Context / Environmental Constraints (can be different sections or combined into one) |
|  | 35 | Functional Requirements, including use-case diagram and description of each use case. |
|  | 7.5 | Software Qualities and Non-functional Requirements |
|  | 5 (+5) | Other Requirements and Other Items. At least a Glossary of Terms. You can earn up to 5 points Extra Credit if you go beyond Glossary |
|  | 7.5 | Assumptions / Risks (can be different sections or combined into one) |
|  | 7.5 | Priorities / Implementation Phases;  Future Directions and Expected Changes |
|  | **100** | TOTAL |

xxx System Requirements

###### July 10, 2013

Peter Anteater

##### Introduction

Hey look, everyone, I’m writing a Requirements Document. And it isn’t so bad after all. Yes, I guess I should have started on it a bit earlier, but at least I didn’t wait until *too* late. Now, let me see, what should go in this section? Introduce the document and the system. Consider the primary stakeholder’s “pains and frustrations” that led to commissioning this system. You can say a few words about the document structure and organization.

Make sure you change the footer to have **your** name in it and not Peter Anteater. Go to View | Header and Footer.

##### Overview / Executive Summary

This section's audience is the non-technical customer and user. Think of a busy executive with little time and little patience to read more than one page. It should address the goals of the system and why it is needed. Describe the major features of the system and the rationale for each (high level only, you will provide details later). Identify and describe other software, processes, hardware, people, and policies that the system may or will affect. List any assumptions made about the existing world or application context (high level only, you will provide details later).

##### Application Context / Environmental Constraints

Now you can provide more detail about “context”, for example will the application run in a business or office, home or outdoors, on a desktop, laptop, tablet, or mobile phone? Also discuss anything you might know about the operating system and “platform”, hardware constraints, design constraints (for example, what should the UI look like), software constraints (such as programming language) etc. Also mention any other software that this software will interact with.

##### Functional Requirements

This section is the heart of the specification document. It clearly describes the proposed software product, including its capabilities and attributes. Do *not* describe details of the user interface, such as “button in the upper left corner” or “press Ctrl-S to save” or “choose from a drop down box”. Instead, focus on the functionality that the system will offer.

Describe the functional requirements of the system in precise detail. When possible, identify the entities (components, sections, areas of functionality) that make up the system. Characterize the properties, states, functions, and interrelationships of each entity. Since this section is the core of the requirements document, it warrants its own brief introduction. Pay careful attention to how this section is organized. Organize this section as follows:

* Very brief introduction to this section
* One use-case diagram showing all actors, use cases, and arrows for relationships
* Text describing each use case (can either be written in regular paragraph form, or as more of a use case template form, describing basic/alternative/exception flows, or a combination of both—see samples for examples of each). **Please organize this section by user, to make it easier to grade. Remember that not all use cases will have alternative and/or exception flows.**

##### Software Qualities and Non-functional Requirements

Generally speaking, software qualities are everything we want our software to be that is not a feature or functional requirement. The categories are usually multi-syllable words that end with “ility” or at least “ty”, such as safety, security, reliability, portability, and especially usability. Discuss constraints pertaining to those qualities, if relevant, as well as speed, space, robustness, implementation bias, etc. Your audiences here are the system designers and programmers, who will probably not be in direct communication with the users. This section will help them assess trade-offs in the system's implementation. List at least three software qualities and/or non-functional requirements.

##### Other Requirements

We often feel we have more to say, in other words somehow the previous two sections did not cover all requirements. This is the place to do it! Any requirements that you are aware of can be described here; you can also add diagrams or other visuals that did not fit in previous sections. At the very least you should define a Glossary of Terms with at least three terms defined.

##### Assumptions / Risks

This is the place to cover any assumptions not covered earlier. Also, list any known risks to the project from any and all perspectives, for example financial risks, business risks, legal and ethical risks, project management and development risks, etc. List at least three assumptions/risks in this section.

##### Priorities / Implementation Phases

If applicable, and following the request of the user, identify one or more subsets of the system's functionality which can or will be implemented first. One way to list priorities is to partition the requirements or use cases into “Must Have (first)”, “Should Have (second)”, and “Nice to Have (third)”.

##### Future Directions and Expected Changes

Again you are providing insight and guidance to the system designers and programmers. List at least four future directions/expected changes.