Software Requirements Specification

for

<Project>

Version 1.0 approved

Prepared by <author>

<organization>

<date created>

Table of Contents

CHANGE HISTORY

DATE	SECTION CHANGED	CHANGE DESCRIPTION
18.06.2017	All Sections	Initial version of SRS document
25.06.2017	Functional & Non-functional Requirements	Incorporated feedbacks given by Prof A. K. Behera
02.07.2017	Env. Requirement Section & Structure chart	Updated info on H/W & S/W platform & added structure chart

1. Introduction

1.1 Purpose

<This section should describe where the software would be deployed and how the software would be used>

1.2 Project Scope

< This section describes the software very briefly, and notes if it is the whole system or part of a larger system. Very short.>

1.3 Environmental characteristics

<Briefly outline the environment (hardware and other software) with which the software will interact. Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

1.4 Definitions, Acronyms and Abbreviations

1.5 References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

2. Overall Description

2.1 Product perspective

<Briefly state as to whether the s/w is intended to be a replacement for a certain existing system, or it is a new software.>

2.2 User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user

classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

2.3 Operating Environment

< Discuss in some detail the hardware on which the software would run, the operating system and other application s/w with which the developed s/w would interact >

2.4 Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software).>

You may have some or None. Think about it though, if you write "none", but it is obvious you should have some, I will deduct points!

2.5 User documentation

<List out the types of user documentation, such as user manuals, on-line help, and trouble shooting manuals that will be derived to the customer alon with the software.>

3. Functional Requirements

<classify the functionalities either based on the specific functionalities involved by different users, or the functionalities that are available in different modes.

i. User class 1
   a. functional requirement 1.1
   b. functional requirement 1.2

ii. User class 2
   a. functional requirement 2.1
   b. functional requirement 2.2

4. External Interface Requirements

SKIP ALL OF SECTION 3 for the SRS… We'll do this as part of the GUI Prototype

4.1 User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style

guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

Think about this in terms of standards, not specific features. For example,

- *all buttons will have a black border*
- *all fonts will be Arial*
- *Draw a screen template showing "main area", "menu here", "status bar" and describe each component. If you have multiple screen layouts depending on the user's current task/settings, describe them*
- *What screen resolutions will you support?*
- *Will you be Section 508 compliant? Are there any other standards you support?*
- *etc…*

4.2 Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

If you system doesn't include hardware, then you'll have none. If it has hardware components, then you should describe (at a high level) how you interface with that hardware.

4.3 Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

*These are internal connections to things like databases, web servers. You mainly need to explain that you have them, but (for CS421) I don't expect detailed information about how you actually connect to them and use them. Just explain that you **do** connect to them and use them for storage of customer information, or to process incoming web requests, etc… These are internal components of your system.*

4.4 Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

These are external communication mechanisms. Do you connect to a bank computer to verify credit card information? That is NOT part of your system, so it is an external communication you have. Describe it here. Do you have other systems connecting in to yours to perform some function? That would also go here.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

5.2 Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

In this section, just say "See section 7 requirements 25-32". And I'll assume those requirements are Safety related.

5.3 Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

In this section, just say "See section 7 requirements 35-42". And I'll assume those requirements are Security related.

5.4 Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

In this section, just say "See section 7 requirements 55-62". And I'll assume those requirements are Software Quality related.

Appendix A: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Don't do any of these for CS421 SRS. You will create these models during the high level design deliverable.

Appendix B: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>

List here any open questions or things you know still need to be done to the SRS, but haven't been addressed yet. (It's okay to have things like that, especially in this CS421 project because we don't have time to do everything.)

CONCLUSION
