Software Requirements

Specification

for

CSE 327.1.4

Version 1.0 approved

Prepared by:

* Emon Sarker
* Labib Rahman
* Parinda Rahman
* Minhazul Abedin Ayaz

14/02/2022

**Table of Contents**

Table of Contents...........................................................................................................................ii

Revision History.............................................................................................................................ii

1. **Introduction**
   1. Purpose
   2. Document Conventions
   3. Intended Audience and Reading Suggestions
   4. Product Scopes
   5. References
2. **Overall Description**
   1. Product Perspective
   2. Product Functions
   3. User Classes and Characteristics
   4. Operating Environment
   5. Design and Implementation Constraints
   6. User Documentation
   7. Assumptions and Dependencies
3. **External Interface Requirements**
   1. User Interfaces
   2. Hardware Interfaces
   3. Software Interfaces
   4. Communication Interfaces
4. **System Features**
   1. System Feature 1
   2. System Feature 2 (and so on)
5. **Other Non Functional Requirements**
   1. Performance Requirements
   2. Safety Requirements
   3. Security Requirements
   4. Software Quality Attributes
   5. Business Rules
6. **Other Requirements**

**Appendix A: Glossary**

**Appendix B: Analysis Model**

**Appendix C: To Be Determined List**

**Parinda-Green**

**Emon-Blue**

**Labib-Purple**

**Minhaz- red**

**1. Introduction**

**1.1 Purpose**

*<Identify the product whose software requirements are specified in this document, including the*

*revision or release number. Describe the scope of the product that is covered by this SRS,*

*particularly if this SRS describes only part of the system or a single subsystem.>*

**1.2 Document Conventions**

*<Describe any standards or typographical conventions that were followed when writing this SRS,*

*such as fonts or highlighting that have special significance. For example, state whether priorities*

*for higher-level requirements are assumed to be inherited by detailed requirements, or whether*

*every requirement statement is to have its own priority.>*

**1.3 Intended Audience and Reading Suggestions**

*<Describe the different types of reader that the document is intended for, such as developers,*

*project managers, marketing staff, users, testers, and documentation writers. Describe what the*

*rest of this SRS contains and how it is organized. Suggest a sequence for reading the document,*

*beginning with the overview sections and proceeding through the sections that are most pertinent*

*to each reader type.>*

**1.4 Product Scope**

*<Provide a short description of the software being specified and its purpose, including relevant*

*benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a*

*separate vision and scope document is available, refer to it rather than duplicating its contents*

*here.>*

**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, use case*

*documents, or a vision and scope document. Provide enough information so that the reader could*

*access a copy of each reference, including title, author, version number, date, and source or*

*location.>*

*Software Requirements Specification for <Project> Page 2*

**2. Overall Description**

**2.1 Product Perspective**

<Describe the context and origin of the product being specified in this SRS. For example, state

whether this product is a follow-on member of a product family, a replacement for certain existing

systems, or a new, self-contained product. If the SRS defines a component of a larger system,

relate the requirements of the larger system to the functionality of this software and identify

interfaces between the two. A simple diagram that shows the major components of the overall

system, subsystem interconnections, and external interfaces can be helpful.>

**2.2 Product Functions**

<Summarize the major functions the product must perform or must let the user perform. Details will

be provided in Section 3, so only a high level summary (such as a bullet list) is needed here.

Organize the functions to make them understandable to any reader of the SRS. A picture of the

major groups of related requirements and how they relate, such as a top level data flow diagram or

object class diagram, is often effective.>

**2.3 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.>

**2.4 Operating Environment**

<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.>

**2.5 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).>

**2.6 User Documentation**

<List the user documentation components (such as user manuals, on-line help, and tutorials) that

will be delivered along with the software. Identify any known user documentation delivery formats

or standards.>

Software Requirements Specification for <Project> Page 3

**2.7 Assumptions and Dependencies**

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in

the SRS. These could include third-party or commercial components that you plan to use, issues

around the development or operating environment, or constraints. The project could be affected if

these assumptions are incorrect, are not shared, or change. Also identify any dependencies the

project has on external factors, such as software components that you intend to reuse from

another project, unless they are already documented elsewhere (for example, in the vision and

scope document or the project plan).>

**3. External Interface Requirements**

**3.1 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.>

A first-time user of the app should see the Role Selection screen and depending on the selected role, the user will be redirected to the role-specific login page. Users can log in using the email used to sign in or their google account (which will require NSU ID and a scanned picture).

At the bottom of the login page, there should be a Sign-Up button that redirects users to the Sign-Up webpage using the default browser.

Upon successful login, the user should be at the main homepage where they have to fill the required fields to submit a complaint. The fields require information such as the complaint description, who the complaint is against, evidence submission, who should review the complaint.

Color Palette for the GUI consists of the following colors as shown below:



Standard Font Family being used:

****

**3.2 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.>

**3.3 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.>

**3.4 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.>

Software Requirements Specification for <Project> Page 4

**4. System Features**

<This template illustrates organizing the functional requirements for the product by system

features, the major services provided by the product. You may prefer to organize this section by

use case, mode of operation, user class, object class, functional hierarchy, or combinations of

these, whatever makes the most logical sense for your product.>

**4.1 System Feature 1**

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

**4.1.1 Description and Priority**

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

**4.1.2 Stimulus/Response Sequences**

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

**4.1.3 Functional Requirements**

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:REQ-2:

**4.2 System Feature 2 (and so on)**

**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.>

Software Requirements Specification for <Project> Page 5

**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.>

**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.>

**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.>

* The internal architecture of the software must be clean and simple so that if the need to change/test/maintain the software arises, it can be done with ease.
* Software must ensure correctness, that is, it should perform the necessary tasks correctly and without any instance of bugs and errors.
* Ease of use is another necessary attribute as the software should have an interface that is clean and easy to use without having to do extensive research.
* The software must be efficient in performing its required task. It should be able to complete its required tasks with the available hardware in an optimum amount of time.
* Maintainability: Each version of the software should be easy to maintain. The code should be structured and clean so that new code can be easily added to the software
* Interoperability: The software must be able to communicate with other solutions with ease.

**5.5 Business Rules**

<List any operating principles about the product, such as which individuals or roles can perform

which functions under specific circumstances. These are not functional requirements in

themselves, but they may imply certain functional requirements to enforce the rules.>

**6. Other Requirements**

<Define any other requirements not covered elsewhere in the SRS. This might include database

requirements, internationalization requirements, legal requirements, reuse objectives for the

project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and

abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire

organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams,

state-transition diagrams, or entity-relationship diagrams.>

Software Requirements Specification for <Project> Page 6

Appendix C: 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.>