Software Requirements Specification

for

<Project>

Version 1.0 approved

Prepared by <author>

<organization>

<date created>

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

In this SRS, we will be presenting our product, a system related to the making and taking of tests. This will describe requirements for the entire system.

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

This SRS may be read by all types of users related to this product. We will describe the many functions and characteristics the completed software system should have. The introduction, as its name implies, just has the objective of introducing a reader to the product. The overall description starts getting a bit more specific in what the product does and requires. The external interface requirements lists requirements that are more related to how the product interacts with the user. System requirements will go deeper into exactly what functions should our system have.

1.4 Product Scope

The product whose software requirements we're specifiying in this document is a system that enables our client to create and modify tests, and share them with users so that they make take them.

2. Overall Description

2.1 Product Perspective

Our product is a new and self-contained one. We plan to make this system from scratch.

2.2 Product Functions

The functions our product must perform are, broadly, letting its owner create, modify and delete tests, and share them with users that might complete them. When the answers to the test are submitted, an appropiate result relevant to the user will be returned.

2.3 User Classes and Characteristics

Primarily, there should be three user types:

1- The administrator, administrators should be able to modify the test selection as needed. Administrators only have privileges regarding tests and their contents.

2- The regular user, users should be able only to take the test and receive the appropiate results. They have the least privileges, and might not even need to have an account.

3- The manager/s of the system, managers are administrators that can also appoint other administrators or remove them, as well as perform rollbacks. Managers are the most important user class

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.>

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).>

We are making this product as part of an educational program, so we lack experience in software developing.

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.>

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.>

4. System Features

4.1 User stories

Following are some user stories that describe the requirements of our system:

Title: Correct Result

As a user, I want my answers to the test to determine the results that most fit me, so that I can know what my answers say about myself.

Priority: High

Title: Result Display

As a user, I want to see my results as percentages on all possible results, so that I can know all the alternatives and how relevant they are to me.

Priority: High

Title: Easy-to-use Interface

As a user, I want the test's graphic interface to be intuitive, so that I can easily answer it.

Priority: Medium

Title: Feedback

As a user, I want to be able to provide feedback, so that I can notify the software's administrators if anything went wrong or I felt my results were inaccurate.

Priority: Low

Title: Test Selection Managing

As an administrator, I want to be able to modify, delete or save a given test, so that I can provide the tests that are needed.

Priority: High

Title: Test Questions Managing

As an administrator, I want to be able to modify, delete or create new questions on a given test, so that I can improve my test as necessary.

Priority: High

Title: Result Managing

As an administrator, I want to be able to decide how much each answer contributes to which result, so that I can provide a more accurate test.

Priority: High

Title: Test Statistics

As an administrator, I want to have statistics related to user answers on my test's questions, so that I can better detect when questions need to be modified.

Priority: Low

Title: Authentification System

As an administrator, I want my system to have a reliable authentification system, so that only people of my choosing can modify, create or delete tests, test questions or results.

Priority: High

Title: Correct Result

As an administrator, I want my tests to be reasonably efficient when processing the answers, so that users don't have to wait long for results.

Priority: Low

Title: Changelog

As a manager, I want to be able to see a history of who modified what, so that if any change is not necessary I can notify the person who did it.

Priority: Medium

Title: Periodic Saves

As a manager, I want my test selection's state to be saved periodically, so that I have backups in case they're needed.

Priority: Low

Title: Rollback capability

As a manager, I want to be able to go back to a previous state of the system, so that if everything else goes wrong I can still go back to a state where the system functioned correctly.

Priority: Low

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.>

5.3 Security Requirements

The product should have an authentication system so that only administrators can modify the test selection. If possible, the manager/s should have higher security, given that their accounts have the most privileges.

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.>

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.>

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.>