**Software Requirements Specification**

**for**

Survival Wisdom

**Version 1.0 approved**

**Prepared by: Loser-Facundo, Marcial-Valentin, Baldebenito-Joaquin**

**Universidad Nacional de Río Cuarto**

**<date created>**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

**2. Overall Description 2**

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

**4. System Features 4**

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

**5. Other Nonfunctional Requirements 4**

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

**6. Other Requirements 5**

**Appendix A: Glossary 5**

**Appendix B: Analysis Models 5**

**Appendix C: To Be Determined List 6**

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

# **Introduction**

## **Purpose**

The purpose of this application is to return the knowledge about survival that we have lost over time with advances in technology. How to act in different types of scenarios and situations. This application will contain practical tips and tests to corroborate the knowledge of each user.

## **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.>*

## **Intended Audience and Reading Suggestions**

*The main objective of the application is for all users to learn about different areas of survival and find out what to do for certain cases in real life. It is suggested that you read the complete guides and perform the tests so that no problems occur that are caused by trying to do something incorrectly.*

## **Product Scope**

*The software is a non-profit project, it will be created with the idea of helping people inform themselves about survival. It consists of five main skills, within which guides are developed where users must read them to learn the content of the skill. Each guide will have a series of questions, as it were, an exam to corroborate their knowledge. The main objective is for people to learn well about survival and if in any case in real life a problem occurs, this knowledge can be applied well and not generate more problems.*

## **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.>*

# **Overall Description**

## **Product Perspective**

*The product was born with the objective of teaching people useful skills in emergency situations in which they have to survive.*

*The context would include emergency situations or natural disasters that users might find themselves in and where the application would be useful in providing knowledge and skills for survival.*

*This product is not the successor of a family of products.*

## **Product Functions**

*The main functions of the product are:*

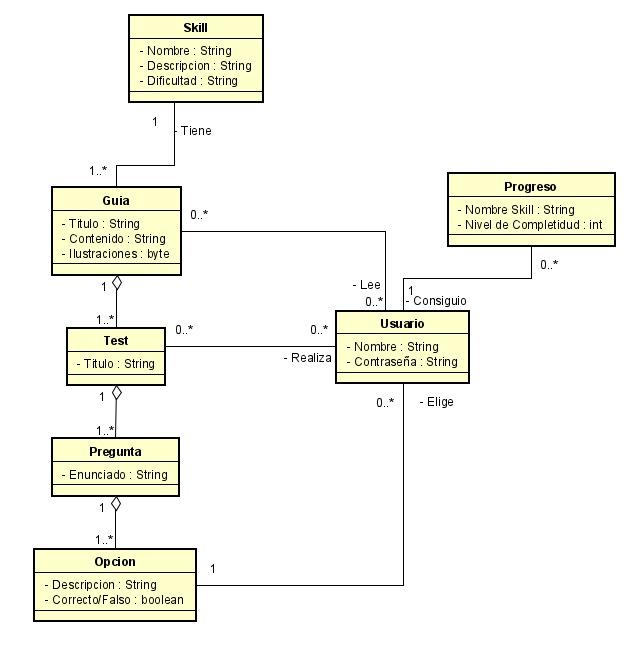
*-register a user.*

*-unsubscribe a user.*

*-read a guide.*

*-take a test from a guide.*

*-view the progress associated with a skill.*

**

## **User Classes and Characteristics**

*-skill class: this has a name, a brief description of what it is about, and a level of difficulty. The difficulty is associated with the learning difficulty and the difficulty of the skill itself.*

*-Guide class: it has a title, the content of the reading and images or illustrations that accompany the reading and facilitate learning. Each guide can have 1 or more tests to verify if the user managed to learn.*

*-test class: it is the test related to a guide and serves to verify how much the user has learned from it. This class has only a title as an attribute.*

*-question class: this has the statement of a question.*

*-option class: has a description that would be the answer shown in some test and a correct/false attribute that is whether that description matches the correct answer to a question.*

*-user class: this has the name of the user's account and the password of the account.*

*-progress class: this has the name of the skill to which the progress belongs and has the level of complexity of said skill. Progress is associated with a user and a skill.*

## **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.>*

## **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).>*

## **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.>*

*ACA HACER MANUAL DE USUARIO!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!*

## **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).>*

*//BASICAMENTE: VAMOS A USAR PROGRAMAS O SOFTWARE EXTERNO PARA LA APLICACIÓN?*

# **External Interface Requirements**

## **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.>*

## **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.>*

## **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.>*

## **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.>*

# **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.>*

## Skill Management

4.1.1 Description

*Users can choose the available survival skills and view the detailed information about a specific skill, including its description and purpose.*

4.1.2 Stimulus/Response Sequences

*You can click the skill icon and you will be redirected to the selected skill page with its guides, content, illustrations and tests.*

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:

## Learning Management

4.2.1 Description

*Users can access guides with instructions and illustrations for a specific skill, also they can take test to assess their understanding of a specific skill, and track their progress in learning different survival skills*

4.2.2 Stimulus/Response Sequences

*When you choose to take a test a page will appear with a multiple choice test where four options will be provided and only one is true,if you complete the test then it will be checked.*

4.2.3 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:

## Learning Management

4.3.1 Description

*Users can create a new account and login later to that existing account. Also they can view and edit their profile information.*

4.3.2 Stimulus/Response Sequences

*When you enter the app, it will require you to login, if you don't have an account it will ask you to create an account. Also you can click your profile icon to access your profile page with all of your information.*

4.3.3 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:

# **Other Nonfunctional Requirements**

## **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.>*

## **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.>*

## **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.>*

## **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.>*

## **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.>*

# **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**

*-Skills: Refers to the different techniques and areas of knowledge necessary to survive.*

*-Guides: It is a set of theoretical documents with illustrations that teach us a “skill”.*

*-Test: It is a series of questions to corroborate a “Skill”.*

*-Survivors: They are the users of the application.*

**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**

*1.2  
1.5  
2.4  
2.5  
2.6  
2.7  
3.1  
3.2  
3.3  
3.4  
4  
4.1.3  
4.2.3  
4.3.3  
5.1  
5.2  
5.3  
5.4  
5.5  
6*

*6.B  
6.C*