Skill Access Manager

For Skill Access Manager Application

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 23/10/16 | 1.0 | filled Chapters 1 through 2.1 with SAM-Project information | Daria Seledtsova |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

1.1 Purpose 4

1.2 Scope 4

1.3 Definitions, Acronyms, and Abbreviations 4

1.4 References 4

1.5 Overview 4

2. Overall Description 4

3. Specific Requirements 5

3.1 Functionality 5

3.1.1 <Functional Requirement One> 5

3.2 Usability 5

3.2.1 <Usability Requirement One> 5

3.3 Reliability 5

3.3.1 <Reliability Requirement One> 6

3.4 Performance 6

3.4.1 <Performance Requirement One> 6

3.5 Supportability 6

3.5.1 <Supportability Requirement One> 6

3.6 Design Constraints 6

3.6.1 <Design Constraint One> 6

3.7 On-line User Documentation and Help System Requirements 6

3.8 Purchased Components 6

3.9 Interfaces 6

3.9.1 User Interfaces 7

3.9.2 Hardware Interfaces 7

3.9.3 Software Interfaces 7

3.9.4 Communications Interfaces 7

3.10 Licensing Requirements 7

3.11 Legal, Copyright, and Other Notices 7

3.12 Applicable Standards 7

4. Supporting Information 7

# Introduction

## Purpose

This Software Requirements Specification was created to collect and organize the requirements for the Skill Access Management Application. It includes a thorough description of the expected functionality for the project concerned, as well as the nonfunctional requirements. These are crucial for the purposes of establishing the understanding between the suppliers of the software and the customers, as well as minimizing the risks connected to the misinterpreting customer’s expectations. The document will furthermore provide the basis for costs-estimation and later validation of the results achieved.

## Scope

The requirements specified in the document are to be applied to the SAM Project. The Skill Access Manager is a free-to-use social platform, which provides a common space for people who want to share their knowledge and profit from other people’s expertise.

ACTORS: There are three types of actors: **guests**, **users** and **admins**.

* Guests – visitors, who either do not have a personal account or are not logged into their profile
* Users – registered SAM-users
* Admins – (extends users) initially SAM-developers, later on a support team with enhanced abilities on the website

SUBSYSTEMS: There are four subsystems to the platform: **user system**, **course system**, **dashboard** and **admin console**.

* User system – allows the actors to be identified as users/guests/admins, store personal information relevant to the platform.
* Course system – allows users to create new, search or book available and rate their attended courses.
* Dashboard – gives users an overview of their progress as tutors (depending on given courses) or students (depending on attended courses), scheduled and recommended classes and account balance.
* Admin console – a system only available for the admins. Allows privileged users to moderate courses and users in accordance with the platform policies.

## Definitions, Acronyms, and Abbreviations

Definitions, Acronyms and Abbreviations relevant for the interpretation of the present document can be found in the project Glossary.

## References

|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | **Date** | **Publishing organization** | **Reference** |
| Project Glossary | 23/10/16 | SAM Initiative | https://github.com/Eynorey/SAM/wiki/Project-Glossary |
|  |  |  |  |
|  |  |  |  |

[This subsection provides a complete list of all documents referenced elsewhere in the **SRS**. Identify each document by title, report number if applicable, date, and publishing organization. Specify the sources from which the references can be obtained. This information may be provided by reference to an appendix or to another document.]

## Overview

The following document will provide all the requirement information connected to the SAM-Project. It will introduce the overall description of the project, based on the vision and a use case diagram. In order to achieve a high level of specification in defining the requirements, all the functions presented in the diagram are going to be examined separately (see chapter 3.1). Further requirements are going to be divided into the following five categories:

* Usability
* Reliability
* Performance
* Supportability
* Design constraints

All the externally obtained components, relevant interfaces and licensing requirements as well as legal disclaimers and standards (if applicable) are going to be listed in chapters 3.8 through 3.12. Chapter 4 will provide the reader with the supporting information for the present document.

# Overall Description

## Vision

Long before the digital era, the world has come to realize that knowledge is the key to any type of development. However, by the end of the 20th century the essence of knowledge itself has been changed dramatically. The new level of information accessibility made it into common property rather than a privilege. The fact, that a kid with a smartphone is no worse than a walking encyclopedia, caused a focus shift from information to skill.

Today, Internet can offer a broad variety of information, but fails to teach you how to apply it. Acquiring practical skills is therefore still associated with a new expense item in the budget and having to leave the comfort of your own home. Our platform is designed to change that.

Skill Access Manager, or SAM, will become a network connecting you to those who are ready to share practical knowledge in hundreds of different spheres and those, who want to learn exactly what you have to offer. Earn points by helping your peers out with whatever you like doing most, whether it is playing piano, skating, or solving mathematical equations. Get rated by your students and invest the points earned to learn from others. It is really as simple as that.

We at SAM believe that everyone has some skills to offer. Something you do as a hobby, or without even thinking about it twice, is what others are dying to learn. Use that to become the better version of yourself. Acquire new skills and share them with others. Smartify yourself. Today. With SAM.

## Use Case Diagram

<INSERT UML>

# Specific Requirements

[This section of the **SRS** contains all software requirements to a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements. When using use-case modeling, these requirements are captured in the Use Cases and the applicable supplementary specifications. If use-case modeling is not used, the outline for supplementary specifications may be inserted directly into this section, as shown below.]

## Functionality

[This section describes the functional requirements of the system for those requirements that are expressed in the natural language style. For many applications, this may constitute the bulk of the **SRS** package and thought should be given to the organization of this section. This section is typically organized by feature, but alternative organization methods may also be appropriate; for example, organization by user or organization by subsystem. Functional requirements may include feature sets, capabilities, and security.

Where application development tools, such as requirements tools, modeling tools, and the like, are employed to capture the functionality, this section of the document would refer to the availability of that data, indicating the location and name of the tool used to capture the data.]

### <Functional Requirement One>

[The requirement description.]

## Usability

[This section includes all those requirements that affect usability. For example,

* specify the required training time for a normal users and a power user to become productive at particular operations
* specify measurable task times for typical tasks or base the new system’s usability requirements on other systems that the users know and like
* specify requirement to conform to common usability standards, such as IBM’s CUA standards Microsoft’s GUI standards]

### <Usability Requirement One>

[The requirement description goes here.]

## Reliability

[Requirements for reliability of the system should be specified here. Some suggestions follow:

* Availability—specify the percentage of time available ( xx.xx%), hours of use, maintenance access, degraded mode operations, and so on.
* Mean Time Between Failures (MTBF) — this is usually specified in hours, but it could also be specified in terms of days, months or years.
* Mean Time To Repair (MTTR)—how long is the system allowed to be out of operation after it has failed?
* Accuracy—specifies precision (resolution) and accuracy (by some known standard) that is required in the system’s output.
* Maximum Bugs or Defect Rate—usually expressed in terms of bugs per thousand lines of code (bugs/KLOC) or bugs per function-point( bugs/function-point).
* Bugs or Defect Rate—categorized in terms of minor, significant, and critical bugs: the requirement(s) must define what is meant by a “critical” bug; for example, complete loss of data or a complete inability to use certain parts of the system’s functionality.]

### <Reliability Requirement One>

[The requirement description.]

## Performance

[The system’s performance characteristics are outlined in this section. Include specific response times. Where applicable, reference related Use Cases by name.

* Response time for a transaction (average, maximum)
* Throughput, for example, transactions per second
* Capacity, for example, the number of customers or transactions the system can accommodate
* Degradation modes (what is the acceptable mode of operation when the system has been degraded in some manner)
* Resource utilization, such as memory, disk, communications, and so forth.

### <Performance Requirement One>

[The requirement description goes here.]

## Supportability

[This section indicates any requirements that will enhance the supportability or maintainability of the system being built, including coding standards, naming conventions, class libraries, maintenance access, and maintenance utilities.]

### <Supportability Requirement One>

[The requirement description goes here.]

## Design Constraints

[This section indicates any design constraints on the system being built. Design constraints represent design decisions that have been mandated and must be adhered to. Examples include software languages, software process requirements, prescribed use of developmental tools, architectural and design constraints, purchased components, class libraries, and so on.]

### <Design Constraint One>

[The requirement description goes here.]

## On-line User Documentation and Help System Requirements

[Describes the requirements, if any, for o-line user documentation, help systems, help about notices, and so forth.]

## Purchased Components

[This section describes any purchased components to be used with the system, any applicable licensing or usage restrictions, and any associated compatibility and interoperability or interface standards.]

## Interfaces

[This section defines the interfaces that must be supported by the application. It should contain adequate specificity, protocols, ports and logical addresses, and the like, so that the software can be developed and verified against the interface requirements.]

### User Interfaces

[Describe the user interfaces that are to be implemented by the software.]

### Hardware Interfaces

[This section defines any hardware interfaces that are to be supported by the software, including logical structure, physical addresses, expected behavior, and so on.]

### Software Interfaces

[This section describes software interfaces to other components of the software system. These may be purchased components, components reused from another application or components being developed for subsystems outside of the scope of this **SRS** but with which this software application must interact.]

### Communications Interfaces

[Describe any communications interfaces to other systems or devices such as local area networks, remote serial devices, and so forth.]

## Licensing Requirements

[Defines any licensing enforcement requirements or other usage restriction requirements that are to be exhibited by the software.]

## Legal, Copyright, and Other Notices

[This section describes any necessary legal disclaimers, warranties, copyright notices, patent notices, wordmark, trademark, or logo compliance issues for the software.]

## Applicable Standards

[This section describes by reference any applicable standard and the specific sections of any such standards which apply to the system being described. For example, this could include legal, quality and regulatory standards, industry standards for usability, interoperability, internationalization, operating system compliance, and so forth.]

# Supporting Information

[The supporting information makes the **SRS** easier to use. It includes:

* Table of contents
* Index
* Appendices

These may include use-case storyboards or user-interface prototypes. When appendices are included, the **SRS** should explicitly state whether or not the appendices are to be considered part of the requirements.]