# Software Requirements Specification

# for

# Book Tracker

Version 0.1 approved

Prepared by <authors>

<team name>

<date created>

Table of Contents

1. Introduction [1](#__RefHeading___Toc441230972)

1.1 Purpose [1](#__RefHeading___Toc7234_2709216707)

1.2 Document Conventions [1](#__RefHeading___Toc441230974)

1.3 Intended Audience and Reading Suggestions [1](#__RefHeading___Toc441230975)

1.4 Product Scope [1](#__RefHeading___Toc441230976)

1.5 References [1](#__RefHeading___Toc441230977)

2. Overall Description [2](#__RefHeading___Toc441230978)

2.1 Product Perspective [2](#__RefHeading___Toc441230979)

2.2 Product Functions [2](#__RefHeading___Toc441230980)

2.3 User Classes and Characteristics [2](#__RefHeading___Toc441230981)

2.4 Operating Environment [2](#__RefHeading___Toc441230982)

2.5 Design and Implementation Constraints [2](#__RefHeading___Toc441230983)

2.6 User Documentation [3](#__RefHeading___Toc441230984)

2.7 Assumptions and Dependencies [3](#__RefHeading___Toc441230985)

3. External Interfaces and Requirements [3](#__RefHeading___Toc441230986)

3.1 User Interface [3](#__RefHeading___Toc441230987)

3.2 Software Interfaces [3](#__RefHeading___Toc441230989)

3.3 Communications Interfaces [3](#__RefHeading___Toc441230990)

4. System Requirements [4](#__RefHeading___Toc441230991)

4.1 Functional Requirements [4](#__RefHeading___Toc441230992)

4.1.1 FREQ-1: [4](#__RefHeading___Toc8213_2709216707)

4.1.2 FREQ-2: [4](#__RefHeading___Toc8215_2709216707)

5. System Scenarios [5](#__RefHeading___Toc441230993)

5.1 Use-case Diagrams [5](#__RefHeading___Toc7236_2709216707)

5.2 Scenarios [5](#__RefHeading___Toc7238_2709216707)

5.2.1 SCN-1: <Scenario name (a sentence)> [5](#__RefHeading___Toc7240_2709216707)

5.2.2 SCN-2: <Scenario name (a sentence)> [6](#__RefHeading___Toc7242_2709216707)

6. System Constraints [7](#__RefHeading___Toc441230994)

6.1 Important Nonfunctional Requirements [7](#__RefHeading___Toc7244_2709216707)

6.1.1 NFREQ-1: [7](#__RefHeading___Toc8217_2709216707)

6.1.2 NFREQ-2: [7](#__RefHeading___Toc8219_2709216707)

7. Other Requirements [7](#__RefHeading___Toc441231000)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Date | Release Description | Version |
| **Felix Friedrich** | 05.07.24 | Template for Software Engineering Course in ETHZ. | 0.3 |
|  | 05.07.24 | <Write your new release description over here> |  |

# Introduction

## 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.>*

## 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

*<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.>*

## 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.>*

## 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

*<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.>*

## 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, can be effective.>*

## 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. >*

## 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.>*

## 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).>*

# External Interfaces and Requirements

## User Interface

*<Describe the logical characteristics of the 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.>*

## 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 Requirements

*<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.>*

## Functional Requirements

*<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind. These are the software capabilities that must be present in order for the user to carry out the services provided by the system.>*

### FREQ-1:

The system should ...

**Description:** *<Describe the requirement >*

**User Priority(x/5):** *< How important is this requirement for the user, comparing with others?>*

**Technical Priority(y/5)**: *<How important is this requirement for the system, e.g. do requirements also need it?>*

### FREQ-2:

...

# System Scenarios

## Use-case Diagrams

*<In this section provide your use-case diagrams of your system. It should be at least one. If there is not enough space inside on the page or inside the diagram, break it in two. This split is also beneficial to group the various scenarios. It is also recommended to keep each scenario and its dependent scenarios in one diagram.>*

## Scenarios

*<Scenario descriptions. In the scenario descriptions, you can find yourselves writing huge amount of text for describing simple things. That is however not what somebody reading this text wants. Be “Concise and meaningful”. Try to link with other points of the document as much as possible. This will help you minimizing the volume of text. If you find yourself specifying more scenarios than asked, that is good, however try to limit yourself from trivial scenarios. A trivial scenario is for example: ”The user types in the console and the console shows the letters the user types.”.>*

### SCN-1: <Scenario name (a sentence)>

|  |  |
| --- | --- |
| FREQ reference | *<Refer to the Functional Requirement numbers this scenario uses.>* |
| NFREQ reference | *<Refer to the Non Functional Requirement numbers this scenario uses. If none then write NA.>* |
| Short Description: | *<A short scenario description. Up to three lines.>* |
| Activation action: | *<What activates this scenario. For example: “Does the user push a certain button?”. Up to two lines.>* |
| Precondition: | *<What is the precondition (state) for this scenario. Keep in mind that different precondition with the same activation action can lead to different scenario. Up to three lines>* |

|  |  |  |  |
| --- | --- | --- | --- |
| Basic flow: <Scenario name> | | | |
| Step | User action | | System response |
| 1 | *<User activation action>* | | *<System does something that gives output to the user>* |
| 2 | *<User action 3>* | | *<This can left empty>* |
| 3 | *<User action 2>* | | *<System does something>* |
| 4 | <This can also left empty> | | *<System does something else>* |
| Post-condition: | | *<Specify the post-condition of this scenario. Hint: It can be the same as the precondition with or without something additional.>* | |

Scenario Diagram for SCN-1 <Scenario name>



### SCN-2: <Scenario name (a sentence)>

*<Your second scenario…>*

# System Constraints

## Important Nonfunctional Requirements

*<If there are nonfunctional 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 system limitations or extensions to various platforms(e.g. cloud servers), safety or security requirements. Make such requirements as specific as possible. You may need to state the connection between functional requirements.>*

### NFREQ-1:

The system should ...

**Description:** *<Describe the non functional requirement >*

**User Priority(x/5):** *< How important is this requirement for the user, comparing with others>*

**Technical Priority(y/5)**: *<How important is this requirement for the system, e.g. do requirements also need it?>*

### NFREQ-2:

*...*

# 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.>*