Software Requirements Specification (SRS)

Submission Date: 21/09/23 - Thursday

A software requirements specification (SRS) is a document that describes **what the software will do and how it will be expected to perform**. It also describes **the functionality the product needs to fulfill all stakeholders (business, users) needs**.

An SRS can be simply summarized into four Ds:

- Define your product's purpose.
- Describe what you're building.
- Detail the requirements.
- Deliver it for approval.

We want to DEFINE the purpose of our product, DESCRIBE what we are building, DETAIL the individual requirements, and DELIVER it for approval. A good SRS document will define everything from, how software will interact when embedded in hardware, to the expectations when connected to other software.

An SRS gives you a complete picture of your entire project. It provides a single source of truth that every team involved in development will follow. It is your plan of action and keeps all your teams — from development to maintenance — on the same page.

How to Write an SRS Document ?

Outline of SRS Document:

1. Introduction

1.1 Purpose

1.2 Intended Audience

1.3 Scope

1.4 Definitions and Acronyms

2. Overall Description

2.1 User Needs

2.2 Assumptions and Dependencies

3. System Features and Requirements

3.1 Functional Requirements

3.2 Nonfunctional Requirements

1. Introduction

1.1 Define the Purpose

Define your Product’s Purpose.

1.2 Intended Audience

Define who in your organization will have access to the SRS and how they should use it. This may include developers, testers, and project managers. It could also include stakeholders in other departments, including leadership teams, sales, and marketing. Defining this now will lead to less work in the future.

1.3 Product Scope

What are the benefits, objectives, and goals we intend to have for this product? This should relate to overall business goals, especially, if teams outside of development will have access to the SRS.

1.4 Definitions and Acronyms

Clearly define all key terms, acronyms, and abbreviations used in the SRS. This will help eliminate any ambiguity and ensure that all parties can easily understand the document.

2. Overall Description

Your next step is to give a description of what you’re going to build. Is it a new product? Is it an add-on to a product you’ve already created? Is this going to integrate with another product? Why is this needed? Who is it for?

2.1 User Needs

Describe who will use the product and how. Understanding the user of the product and their needs is a critical part of the process.

Who will be using the product? Are they a primary or secondary user? Do you need to know about the purchaser of the product as well as the end user? In medical devices, you will also need to know the needs of the patient.

2.2 Assumptions and Dependencies

What are we assuming will be true? Understating and laying out these assumptions ahead of time will help with headaches later. Are we assuming current technology? Are we basing this on a Windows framework? We need to take stock of these assumptions to better understand when our product would fail or not operate perfectly.

3. System Features and Requirements

In order for your development team to meet the requirements properly, we must include as much detail as possible.

3.1 Functional Requirements

Functional requirements of a product means the functionality provided by the product/system. **Functionality** of the product describes the features, capabilities, and limitations or constraints that might affect the software's performance.

Asking yourself the question "does this add to my tool's functionality?" Or "What function does this provide?" can help with this process.

You may also have requirements that outline how your software will interact with other tools, which brings us to external interface requirements.

3.2 Nonfunctional Requirements

Non-functional requirements state constraints on the design and construction of the product. Nonfunctional requirements, which help ensure that a product will work the way users and other stakeholders expect it to, can be just as important as functional ones. While a system can still work if non functional requirements are not met, it may not meet user or stakeholder expectations, or the needs of the business.

These may include:

Security : Does your product store or transmit sensitive information? Does your IT department require adherence to specific standards? What security best practices are used in your industry?

Capacity : What are your system's storage requirements, today and in the future? How will your system scale up for increasing data storage volume demands?

Compatibility : What are the minimum hardware requirements? What operating systems and their versions must be supported?

Reliability and Availability : What is the critical failure time under normal usage? Does a user need access to this all hours of every day?

Maintainability and Manageability : How much time does it take to fix components, and how easily can an administrator manage the system? Under this umbrella, you could also define Recoverability and Serviceability

Scalability requirements: How easy is it to use the product? What defines the experience of using the product/

Scalability requirements: What are the highest workloads under which the system will still perform as expected?