System Requirements Specification

version x.x

for

<Project Name>

prepared by

|  |  |
| --- | --- |
| name | email |
|  |  |

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1. Introduction](#_tgznxvlaqu3f) 3

[1.1. Purpose](#_leol45ocan48) 3

[1.2. Scope](#_q220v5kph5bu) 3

[1.3. Definitions, acronyms, and abbreviations](#_3znysh7) 3

[1.4. References](#_2et92p0) 3

[2. Overall description](#_tyjcwt) 4

[2.1. Product perspective](#_1t3h5sf) 4

[2.2. Product functions](#_4d34og8) 4

[2.3. User characteristics](#_2s8eyo1) 4

[2.4. Constraints](#_17dp8vu) 5

[2.5. Assumptions and dependencies](#_3rdcrjn) 5

[3. Specific requirements](#_26in1rg) 6

[3.1. External interfaces](#_lnxbz9) 6

[3.2. Functional requirements](#_1ksv4uv) 6

[3.2.1. Actor goal list](#_44sinio) 7

[3.2.2. Use case view](#_gw55xoxk7a9e) 7

[3.3. Non-functional requirements](#_z337ya) 8

[3.3.1. Performance efficiency](#_sac2dz8jeipd) 8

[3.3.2. Compatibility](#_3ttsxj5mvra3) 8

[3.3.3. Usability](#_ta2ye62qwtnv) 8

[3.3.4. Reliability](#_mfzz872z2f96) 8

[3.3.5. Security](#_en5crhns4lin) 8

[3.3.6. Maintainability](#_cirhpvyt8qnz) 8

[3.3.7. Portability](#_rskelybx7obc) 8

[3.3.8. Design constraints](#_oyp1v9cluj58) 8

[3.3.9. (On-line) user documentation and help](#_2jwvmqy6byhq) 9

[3.3.10. Purchased components](#_45vokn5kdv7f) 9

[3.3.11. Licensing requirements](#_neewhext8nr6) 9

[3.3.12. Legal, copyright and other notices](#_rfikm4bnwr9r) 9

[4. Analysis Models](#_3j2qqm3) 10

# 1. Introduction

The introduction of the Software Requirements Specifications Document provides an overview of the entire document.

## 1.1. Purpose

This section defines the role or purpose of the Software Requirements Specifications Document and briefly describes the structure of the document. Identify the intended audience for the document is identified, with an indication of how they are expected to use the document.

## 1.2. Scope

A brief description of what the Software Requirements Specifications Document applies to; what is affected or influenced by this document.

## 1.3. Definitions, acronyms, and abbreviations

Provides the definitions of all terms, acronyms, and abbreviations required to properly interpret the Software Requirements Specifications Document. This information may be provided by reference to the project’s Glossary.

## 1.4. References

Provide a list of all documents referenced in the SRS.

# 2. Overall description

This section describes a background to the requirements: The general factors that affect the product, such as constraints, assumptions and dependencies.

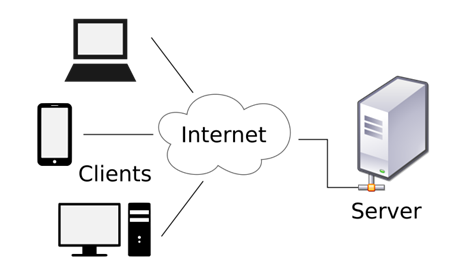


Figure 1. A client-server architecture.

## 2.1. Product perspective

Is the product self-contained? If not, then put the product into perspective with other related products. Use a block diagram to show the big picture.

## 2.2. Product functions

Provide a summary of the major system functions.

## 2.3. User characteristics

Who are the intended users of the system what is their expected educational level, experience and technical expertise?

## 2.4. Constraints

Describe any items that will limit the options of the developers (such as regulations, hardware limitations, safety and security etc.)

## 2.5. Assumptions and dependencies

What assumptions are there? For example, a specific operating system should be present on a given hardware platform. If not, this document would have to be changed.

# 3. Specific requirements

This section contains all requirements in detail: Functional as well as non-functional requirements (quality attributes and constraints). The quality attributes are listed according to the *ISO/IEC 25010* standard that classifies software quality in a structured set of characteristics and sub-characteristics.

## 3.1. External interfaces

A detailed description of all inputs into the system and all outputs from it (in terms of content and form).

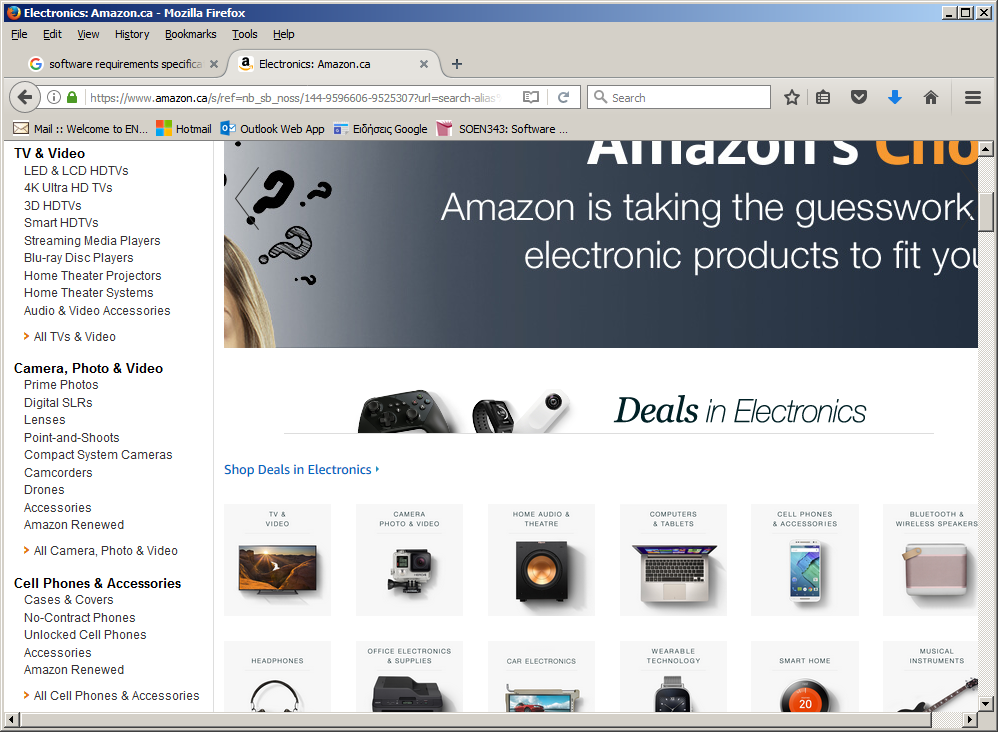


Figure 2. An example user interface..

## 3.2. Functional requirements

Functional requirements capture the intended behaviour of the system. This section contains the *Actor Goal List* and the *Use Case view*.

### 3.2.1. Actor goal list

|  |  |
| --- | --- |
| Actor | Goal |
|  |  |

### 3.2.2. Use case view

The use case model is shown in Figure 3.

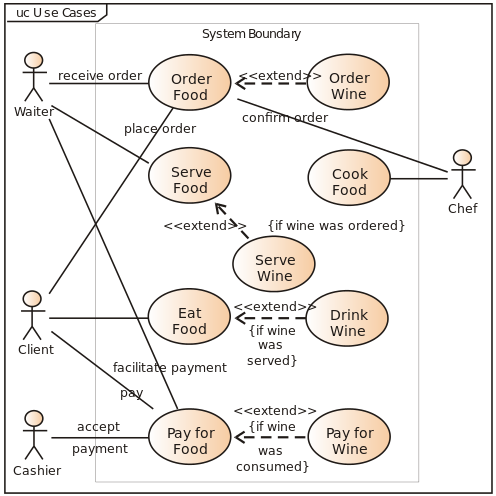


Figure 3. Use case model.

Description of individual use cases goes here.

## 3.3. Non-functional requirements

### 3.3.1. Performance efficiency

Description goes here.

### 3.3.2. Compatibility

Description goes here.

### 3.3.3. Usability

Description goes here.

### 3.3.4. Reliability

Description goes here.

### 3.3.5. Security

Description goes here.

### 3.3.6. Maintainability

Description goes here.

### 3.3.7. Portability

Description goes here.

### 3.3.8. Design constraints

Decisions that must be followed, such as languages, processes, prescribed use of tools, architectural and design constraints, purchased components, class libraries, etc.

### 3.3.9. (On-line) user documentation and help

Description.

### 3.3.10. Purchased components

Description.

### 3.3.11. Licensing requirements

Description.

### 3.3.12. Legal, copyright and other notices

Description.

# 4. Analysis Models

List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.

Illustrate (system) UML sequence diagrams (one for each critical scenario), identify system operations and describe operation contracts, one per critical system operation.

You may also use UML state diagrams to describe critical use cases, one state diagram per use case.

Finally, create a UML conceptual class diagram (“domain model”) for the system. If the model gets too large, you can use UML package diagrams to provide logical grouppings for the model.