Version <1.0>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 23/Mar/17 | <x.x> | <details> | Pop Cristian |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

2. Non-functional Requirements 4

2.1 Availability 4

2.2 Performance 4

2.3 Security 4

2.4 Testability 4

2.5 Usability 4

3. Design Constraints 4

# Introduction

[The introduction of the **Supplementary Specification** provides an overview of the entire document.

The **Supplementary Specification** captures the system requirements that are not readily captured in the use cases of the use-case model. Such requirements include:

Legal and regulatory requirements, including application standards.

Quality attributes of the system to be built, including usability, reliability, performance, and supportability requirements.

Other requirements such as operating systems and environments, compatibility requirements, and design constraints.]

# Non-functional Requirements

## Availability

Availability describes the probability of a system being operational at a given moment. This is described by using metrics like the ‘mean time to failure’ or ‘mean repair time’.

Source of stimulus: it can be internal or external to the system

Stimulus: hardware failure, software crash

Environment: The system running normally

Artifact: Database, application server

Response: replace or reboot the component that caused the crash

Response measure: repair time and mean repair time

* The server should boot up after a crash in under 2 minutes

## Performance

Performance describes the system’s response time.

Source of stimulus: external to the system

Stimulus: requests coming from the users

Environment: server running normally

Artifact: The system

Response: The system processes the received requests

Response measure: delay, miss rate

* 90% of the incoming requests should be processed in under 1 second

## Security

Security describes the system’s ability to deny unauthorized access.

Source of stimulus: malicious users that are external to the system

Stimulus: an unauthorized or malicious request

Environment: online system

Artifact: the system

Response: The system should deny access to unauthorized users and it should filter out invalid or malicious input

Response measure: percentage of unauthorized access allowed

* The system should not be vulnerable to SQL injection
* The system should not allow unauthenticated users to access data
* The system should not allow regular users to access functions and data reserved for administrators

## Testability

Testability describes how easy the software faults can be discovered.

Source of stimulus: internal

Stimulus: unit tests, integration tests

Environment: the application

Artifact: the code of the system’s software

Response: the activity determined by the arrival of the stimulus

Response measure: time to perform tests

* The system should have unit tests for the most important operations

## Usability

Usability describes how easy it is for new users to learn to use the system and how their errors impacts the system.

Source of stimulus: external (the end-users)

Stimulus: invalid user input

Environment: system running normally

Artifact: the system

Response: error messages, invalid input rejected

Response measure: number of errors

* The system should validate all the data coming from the users and reject the invalid data
* The system should display descriptive error messages

# Design Constraints

The system should use the Layers architectural pattern. The technologies involved in creating the server should Tomcat Embedded (Spring Boot) and Spring.