Online Pizza Ordering System

Supplementary Specification

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 20/03/18 | 1.0 | Draft of the first non-functional requirements and design constraints | Andreea Ionutas |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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

Supplementary Specification

# Introduction

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

The system should have high availability, a client being able to access and use the application 98-99,99% of time.  The system should be in an operable and committable state 23-24 hours a day/7 days a week.

## Performance

The system should have high computer performance, which involves a short response time for the requests that clients make. It is also necessary to take into consideration the number of clients accessing the system at the same time. The response time should not increase dramatically them multiple users are placing an order.

## Security

Depending on the type of user using the application, there will be actions specific to each one of them. When logging in as an administrator, the user would have some additional functionalities, such as viewing a sales report.

## Testability

The implemented system will be tested in multiple ways from unit testing to integration tests. The tests will be from controller level to each method implemented.

## Usability

The application must be designed in a user friendly way, such that anyone with an Internet connection and interested in placing an order should be capable of using it. According to the type of user, there will be special functionalities, making distinction between a regular client and an administrator.

# Design Constraints

The application will be implemented in Java language. The main limitations of this program language is the fact that it is kind of slow comparing to other languages and the status of security is uncertain.

For an easier implementation, we will use the Spring framework which can be used for all layer implementations of a real time application. It will be a Maven project. Maven projects are defined with an XML file named pom.xml. Among other things, this file gives the project's name, version, and dependencies that it has on external libraries.

The data will be stored in a database and we will make use of MongoDB. As for the GUI, we would use HTML/CSS and AngularJS.