Gym Management System

Supplementary Specification

Version <1.0>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <dd/mmm/yy> | <x.x> | <details> | <name> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 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.]*

The project’s name is Gym Management System and it is supposed to be an application that can be used inside a gym. The users of this application (mainly the staff of the gym) can create new memberships for new customers, car renew the old memberships and are able to create training programs for customers. Also, the owner of the gym will have the possibility to track the gym’s incomes.

# Non-functional Requirements

*[Define system quality attributes in terms of scenarios according to the following template:*

* *Quality attribute definition*
* *Source of stimulus: the entity (human or another system) that generated the stimulus or event*
* *Stimulus: a condition that determines a reaction of the system*
* *Environment: the current condition of the system when the stimulus arrives*
* *Artifact: is a component that reacts to the stimulus. It may be the whole system or some pieces of it*
* *Response: the activity determined by the arrival of the stimulus*
* *Response measure: the quantifiable indication of the response*
* *Tactics*

*]*

## Availability

## Availability represents the degree to which a system, subsystem or equipment is in a specified operable and commitable state at the start of a specific task / service. Simply, it can be thought of as the proportion of time in which the system is found in a functioning condition. The application should be available to its users at any given time and different failures (such as database server failure) should not affect it’s using process.

## Performance

Performance is the amount of useful work accomplished by a computer system. It can be estimated in terms of accuracy, efficiency, and speed of executing computer program instructions. Our application should generate memberships fast enough because we don’t want our customers to wait for us. Also, the report with the gym’s income should be generated instantly.

## Security

Security represents the capability of a system to prevent attacks, data stealing, accidental actions and also to provide privacy.

## Testability

## Software testability is the degree to which a software artifact (i.e. a software system, software module, requirements- or design document) supports testing in a given test context. If the testability of the software artifact is high, then finding faults in the system (if it has any) by means of testing is easier.

My application will be tested using unit tests, testing the flows that are successful and those who fail.

## Usability

In software engineering, usability is the degree to which a software can be used by specified consumers to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use.

My application will offer a graphical user interface so the operations will be eased.

# Design Constraints

*[This section needs to indicate any design constraints on the system being built. Design constraints represent design decisions that have been mandated and must be adhered to. Examples include software languages, software process requirements, prescribed use of developmental tools, architectural and design constraints, purchased components, class libraries, and so on.]*

The application will be implemented using C# language and maybe I will integrate Entity Framework for simplifying mapping between objects and tables of relational database. For the database storage, I will use SQL Server and for the development environment I will use Visual Studio.