Scrum planner

Supplementary specification

Version 1.1

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 12/03/2016 | 1.0 | Initial draft | Stefan Pamparau |
| 20/03/2016 | 1.1 | Second draft | Stefan Pamparau |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1.Introduction 3](#__RefHeading___Toc2083_1004426867)

[2.Non-functional Requirements 3](#__RefHeading___Toc2085_1004426867)

[2.1Availability 5](#__RefHeading___Toc2087_1004426867)

[2.2Performance 5](#__RefHeading___Toc2089_1004426867)

[2.3Security 5](#__RefHeading___Toc2091_1004426867)

[2.4Testability 6](#__RefHeading___Toc2093_1004426867)

[2.5Usability 6](#__RefHeading___Toc2095_1004426867)

[3.Design Constraints 6](#__RefHeading___Toc2097_1004426867)

# Introduction

The introduction of the current Supplementary Specification documentprovides 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

This section provides definitions for quality attributes for the most important scenarios of the system. These are:

User registration scenario:

-attribute definition: reliability expressed by the fact that the server must be deployed and must correctly and efficiently process the user's request

-source of stimulus: user presses the registration button

-stimulus: the central server receives request from client

-environment: the central server is deployed

-artifact: the central server's dispatcher servlet receives the request and calls associated facades, services, daos to process the request

-response: user is registered into the system

-response measure: indicated by the response code defined in the Htpp protocol specification

User login scenario:

-attribute definition: reliability expressed by the fact that the server must be deployed and must correctly and efficiently process the user's request

-source of stimulus: user presses the login button

-stimulus: the central server receives request from client

-environment: the central server is deployed

-artifact: the central server's dispatcher servlet receives the request and calls associated facades, services, daos to process the request

-response: user is logged into the system

-response measure: indicated by the response code defined in the Htpp protocol specification

User join project scenario:

-attribute definition:

* reliability expressed by the fact that the server must be deployed and must correctly and efficiently process the user's request
* response time

-source of stimulus: user presses the join project button

-stimulus: the central server receives request from client

-environment: the central server is deployed

-artifact: the central server's dispatcher servlet receives the request and calls associated facades, services, daos to process the request

-response: user joins the project

-response measure: indicated by the response code defined in the Htpp protocol specification

User create user story scenario:

-attribute definition:

* reliability expressed by the fact that the server must be deployed and must correctly and efficiently process the user's request
* the response time must be minimal

-source of stimulus: user presses the create user story button

-stimulus: the central server receives request from client

-environment: the central server is deployed

-artifact: the central server's dispatcher servlet receives the request and calls associated facades, services, daos to process the request

-response: user story is created

-response measure: indicated by the response code defined in the Htpp protocol specification

User create user story failure scenario:

-attribute definition:

* fault tolerant

-source of stimulus: user presses the create user story button

-stimulus: the central server receives request from client but the information supplied is incorrect

-environment: the central server is deployed

-artifact: the central server's dispatcher servlet receives the request and calls associated facades, services, daos to process the request

-response: error message is returned to the user

-response measure: indicated by the response code defined in the Htpp protocol specification

## Availability

To maximize the system's market impact, it must be available on all the popular OS's. In order to achieve this requirement an OS independent programming language must be used. Thus, the system will be developed using the Java programming language which is OS independent through the JVM.

The most popular OS's on which the system will be available are:

* Windows
* Linux
* Mac OS

The system's availability can be also described by the total uptime of the central server. To ensure maximum client satisfaction, the central server must be deployed using load balancers, preferably using a cloud solution.

## Performance

Because the system involves many client-server interactions, the performance of the central server is critical. In order to achieve the performance constraints imposed the following measures will be applied:

* the design of the server will be kept to a minimum (avoid over engineering)
* caching layers will be introduced
* a machine with the recommended hardware resources will be used to deploy the server

The desktop client's performance must also be taken into consideration. Although this generally depends on

end user's hardware, the desktop client will be designed to be able to run fluently on the majority of personal computers.

## Security

The Scrum Planner performs requests which contain user passwords or private information belonging to a company. Thus, the system will be secured used one of the available open source frameworks (e.g.: spring security).

## Testability

To ensure further ease of development, test cases will be implemented for classes and cross cutting concerns. An open source testing framework will be used (e.g.: Junit).

## Usability

In order to increase usability, the system must be easy to install, intuitive and easy to use. To meet this requirement special attention must be given when designing the application's UI.

# Design Constraints

In the System Development Life Cycle afferent to phase 1 of the Scrum Planner application multiple design constraints have been imposed. They are listed below together with their solutions:

* an object oriented programming language was chosen: the system will be developed using Java
* IDE must be used during development to speed up process: Intellij IDEA Ultimate was chosen
* a minimum of 3 design patterns must be used
* an agile style of development must be used
* a history of project development must be kept: git will be used as a versioning system. It also keeps a history of the projects development