Travel Agency

Supplementary Specification

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 03.04.2019 | 1.0 | Supplementary specification for the first version of the project | Debre Lóránd - Sándor |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 document aims to present the requirements of the Travel Agency system and captures the system requirements that are not captured in the Use Case model. The application is an online web spring application, that help the user to find the best holiday package, to add the package to chart and buy them, and to reduce as much as possible the search time. The system also allows to see the description and review about every country, city and hotel actually is a collection of holiday package with their review.

For this site we need to have knowledge of Front-end with CSS, JavaScript and HTML, and Back-end including programming language as Java, MySQL.

# Non-functional Requirements

The system should always be connected to database server because the data must be saved in real time, and we need to have access to all information.

This specification defines the non-functional requirements like availability, performance, security and usability.

## Availability

The application will be available on any computer/smart-phone all the time, in 24 hours a day – but there shall be a low percentage of downtime, when the maintenance man can add new features, etc. – but can not exceed 1% of the time.

## Performance

The system shall have no more than 5 seconds of delay, if somebody accessing the database using interface and also if more users want to see the same details the system need to serve them.

## Security

The user can log in, this feature will be ensured by using an encryption algorithm and the crypted user password will be stored in the database, so the passwords are not visible to everyone and can not be decrypted into human language. Every user input will also be verified using a validator, and if it pass the test will be taken into considerations.

## Testability

The application can be tested separately (Back-end/Front-end) on every module with a tool and find defects easy.

## Usability

The system can be used by everyone but only a registered user can book something. There will be a clear distinction between user and administrator (agent). Any user without an account can create an account easily. The application is user friendly with simple navigation, that allows unexperienced users to use it.

# Design Constraints

The system must be implemented using a high-level language: Java. Layers and another architectural patterns should be used with input validation. The application needs to be a client-server architecture, the server has to running when the application is to be run. The spring framework will be used using a Maven configuration.