**Map Your Hike**

**Version <1.0>**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 03/04/19 | 1.0 | basic concepts | Dobner Imola |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**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**

In this section of the documentation I will present you the non-functional requirements of the Map your hike web application. As a web application usability, reliability, performance besides security are the most important aspects. It is required for the application to be compatible with the operating system, and in the case of specific hardware components so be also true.

* **Non-functional Requirements**
* **Availability**
* **Performance**
* **Security**

The application will be secure, using prepared statements for working with the database, creating it and maintaining. It should not be easily attacked by outsiders and keep al the information private about it’s users.

* **Testability**

The appliation's main purposes will be tested in detail, to be sure everything is working as it should, but all in all it will be tested by the users. Considering that each class will have a purpose and just a responsability, hopefully it will manage to work properly.

* **Usability**

This application will provide a source for many informations needed by a hiker, before starting a trip. The best part for the user is that he can access all of it on a single platform, with no need to consult a big variaty of information sources. Also using this application anybody can easily plan a trip to bits, having a tamplate for every aspect that should be taken into count( water supply, accomodation, diffiulty of the trip and so on).

* **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.]*