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 Usability 4

3. Design Constraints 4

# Introduction

This application is a client-server system for a traveling agency.

When running the application, the first frame we see is a login frame for authentication. Here the user has to insert his credentials to login in the application. If the credentials have administrator rights then the administrator frame will be opened, otherwise the employee frame will be shown. In the administrator frame the user can perform the CRUD operations on the users in the database. The employee frame offers some similar features as the administrator frame: CRUD on clients and CRUD on vacations. The employee frame also allows the user to book a vacation for a certain client.

# Non-functional Requirements

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

* Quality attribute definition
* Source of stimulus: human stimulation
* Stimulus: User inserting or requiring data
* Environment: The system is in a waiting state prepared to react to the user
* Artifact: the whole system reacts to a stimulus
* Response: the response is different for each stimulus
* Response measure: the user is informed by the status of his stimulus

## Availability

The application is available for desktop system running on windows, but the fact that it is a client-server application gives the opportunity to extend it on others systems.

## Performance

The application works only with raw data so this give a big advantage to the performance time. The responses are instant and the user has to wait no time after the application.

## Security

The data in database is secured by a login procedure. The login data is also secured by storing the passwords encrypted in the database.

## Usability

Because of its friendly and simple interface the application is easy to use and understand by any type of user.

# Design Constraints

The code for this application was written using .net programming language. Compiler Software – Visual Studio 2013. Architectural Constrains – Client-Server Application. Different Form Application for each type of user. Using a database to store information. Storing passwords in a safe manner.