History Tinder

Supplementary Specification

Version <1.0>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 21/Mar/20 | 1.0 | Conceptual description | Butuza Dan-Radu |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 goal of the application is to create an application similar to Tinder, but with historical characters. Based on the location introduced by the user, the application will start listing a number of personalities from a database. There will be two types of users, normal ones, which will be accessing the application normally, as specified above and admins, which will be able to add new personalities, remove them, or edit the information of a character.

The application will be designed in a friendly manned and will serve an educational purpose, making learning history more interesting through modern means, making it easy to use by anyone. In terms of reliability, the only problem that would arise would be the veracity of the information that will be displayed for a given personality, this is why, bibliography links will be displayed at the bottom of the page from verified sources. The application should be supported by any device with java support and input/output peripherals. Most OS have Java support, therefore, the application should be able to run on them.

# Non-functional Requirements

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Type of requirements | Low | Nominal | High | Very high |
| 1 | Performance requirements | Low |  |  |  |
| 2 | Interface requirements | Low |  |  |  |
| 3 | Operational requirements | Low |  |  |  |
| 4 | Resource requirements | Low |  |  |  |
| 5 | Design requirements and implementation constraints |  | Nominal |  |  |
| 6 | Security and privacy requirements |  | Nominal |  |  |
| 7 | Portability requirements | Low |  |  |  |
| 8 | Software quality requirements |  |  | High |  |
| 9 | Software reliability requirements | Low |  |  |  |
| 10 | Software maintainability requirements | Low |  |  |  |
| 11 | Software configuration and delivery requirements |  |  |  | Very high |
| 12 | Data definition and database requirements |  | Nominal |  |  |
| 13 | Human factor related requirements |  | Nominal |  |  |
| 14 | Adaptation requirements | low |  |  |  |

## Availability

Availability is concerned with system failure and its associated consequences

Source of stimulus: internal, external to the system

Stimulus: crash, timing, unanticipated message

Environment: normal operation at run time, also at design time

Artifact: process, system

Response: deploy/notify modification, continue or not

Response measure: no downtime

## Performance

Response time for a transaction(average, maximum)

Throughput (e.g., transactions per second)

Capacity (e.g., the number of customers or transactions the system can accommodate)

Degradation modes (what is the acceptable mode of operation when the system has been degraded in some manner)

Resource utilization: memory, disk, communications, etc.]

## Security

Security is a measure of the system’s ability to resist unauthorized usage while still providing its services to legitimate users

Source of stimulus: correctly identified individual

Stimulus: tries to modify information

Environment: under normal operations

Artifact: data within the system

Response: system maintains audit trail, blocks the access

Response measure: correct data is restored within a day

## Testability

Testability refers to the ease with which software can be made to demonstrate its faults through testing

Source of stimulus: unit tester/developer

Stimulus: performs unit test

Environment: at the completion of a component

Artifact: component of the system

Response: component has interface for controlling behavior and output of the component is observable

Response measure: path coverage of a specific percentage is achieved

## Usability

Usability refers to how easy is for the user to accomplish a desired task and the kind of user support the system provides

Source of stimulus: users

Stimulus: minimize impact of errors

Environment: at runtime

Artifact: system

Response: wishes to cancel the current operations

Response measure: cancellation takes less than a second

# 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 should be developed in Java and use a layered architecture. The data will be stored in a database.