Love at First Type

Software Requirements Specification (SRS) Document

MUSoftwareRequirementsSpecification.doc

Version 1.0

February 1, 2016

Charming Cupids
<table>
<thead>
<tr>
<th>Version</th>
<th>Primary Author(s)</th>
<th>Description of Version</th>
<th>Date Completed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version 1.0</td>
<td>Nikki Brown, Rebecca Steele, Collin Peters, Jarrett Ruffin</td>
<td>Final Draft</td>
<td>02/01/16</td>
</tr>
</tbody>
</table>
# Table of Contents

Introduction ............................................................................................................................... 5

1.1 Purpose ........................................................................................................................ 5

1.2 Scope ............................................................................................................................ 5

1.3 Overview ...................................................................................................................... 5

1.4 Glossary of Terms ......................................................................................................... 5

2 Overall Description ........................................................................................................... 6

2.1 Product Perspective ...................................................................................................... 6

2.1.1 System Interfaces ................................................................................................... 6

2.1.2 Memory Constraints .............................................................................................. 6

2.1.3 Operations ............................................................................................................. 6

2.2 Product Functions ........................................................................................................ 6

2.3 Similar Systems ............................................................................................................ 6

2.4 User Characteristics .................................................................................................... 6

2.5 User Objectives ........................................................................................................... 6

2.6 Constraints ................................................................................................................... 7

3 Specific Requirements ........................................................................................................ 8

3.1 Functional Requirements .............................................................................................. 8

3.2 User Interface Requirements ........................................................................................ 9

3.2.1 User Interface: Graphical (GUI) or Command-Line (CLI) ...................................... 9

3.2.2 Application Programming Interface (API) .............................................................. 9

3.2.3 Diagnostics (Error Reporting and Usage Logs) ....................................................... 9

3.3 System Requirements .................................................................................................. 9

3.3.1 Hardware Interfaces .............................................................................................. 9

3.3.2 Communications Interfaces ...................................................................................... 9

3.3.3 Software Interfaces ................................................................................................ 9

3.4 Domain Requirements/Constraints .............................................................................. 9

3.5 Non-Functional Requirements ....................................................................................... 9

3.5.1 Reliability .............................................................................................................. 9

3.5.2 Colors and Font Concepts ......................................................................................... 9

3.5.3 Availability ........................................................................................................... 10

3.5.4 Security .................................................................................................................. 10

3.5.5 Maintainability ...................................................................................................... 10
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.6</td>
<td>Logical Database Requirements</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>Software Life Cycle Model</td>
<td>11</td>
</tr>
<tr>
<td>4.1</td>
<td>Choice of Software Life Cycle Model</td>
<td>11</td>
</tr>
<tr>
<td>4.2</td>
<td>Justification for Choice of Model</td>
<td>11</td>
</tr>
</tbody>
</table>
1.1 Purpose
The purpose of this document is to specify specific requirements of this project along with choosing a life cycle model that best suits this project. The intended audience of this document is any Internet user intending on using this project to find love.

1.2 Scope
The software product, Love at First Type, is intended for users to find love, or at least a relationship. It is an online dating service that will allow a user to enter characteristics about themselves as well as characteristics they are interested in finding in a mate. It will then match different users based on desired characteristics.

1.3 Overview
Section two of this document is product perspective. The product perspective includes system interfaces, memory constraints, operations, and functions. Section three includes functional requirements, system requirements, and nonfunctional requirements. Section four is the Software Life Cycle Model chosen for this project, as well as the reason for this choice.

1.4 Glossary of Terms
SLCM - Software Life Cycle Models are a process for planning, creating, testing, and deploying an information system.
GitHub-source-code hosting service. Allowing for multiple people to fork and contribute to one project.
2.1 Product Perspective
This product is independent and self contained.

2.1.1 System Interfaces
The web based interface will communicate with a database server based at the University of Mount Union. The software will be written in C# using Visual Studio 2015.

2.1.2 Memory Constraints
Memory constraints are not a factor.

2.1.3 Operations
The software will run in real time. Each user will get an immediate response from the server. The software source code will be safely stored in a repository on GitHub allowing for easy recovery.

2.2 Product Functions
The main function of this project is for two users to make a love connection. It will allow each user to create a unique profile with their picture and their characteristics. They will also be able to specify the characteristics they are interested in finding in another person. Then the software will pair profiles that have interests that match what other users are looking for.

2.3 Similar Systems
There are many similar systems that exist on the Internet. A few examples are eHarmony, Christian Mingle, and Farmers Only. Our system and these systems have a user establish a profile, then connects them with people with similar interests. However there are differences between all these sites. Sites such as Christian Mingle and Farmers Only have a specific type of user, whereas our product is open to any user who wants to give love a chance.

2.4 User Characteristics
The typical user will not need an extensive technology background. They will need a basic understanding of how to navigate a webpage.

2.5 User Objectives
- Each user can expect to be able to create their own personal profile.
- Each user can expect to receive a list of profiles that are potential prospects for them.
- Each user can expect to be provided with their match’s contact information.
- Each user can expect their list of matches to be saved, therefore every time they login they will still be there.
2.6 Constraints

-Time constraint: our biggest constraint is our time. We have limited time before graduation to add all the functions we want to add to this program.

-Limited current knowledge of team members.
3.1 Functional Requirements

Criticality Scale: Low (1) – Medium (2) – High (3)

1. Create a database to store users’ profiles.
   Criticality: 3
   Dependencies: None

2. Allow users to be able to create individual profiles and enter data about themselves and what they are looking for.
   Criticality: 3
   Dependencies: 1

3. A matching algorithm designed to take the interests of one person and compare against the interests of all the other users and returns a list of users that would be most compatible with the original user.
   Criticality: 3
   Dependencies: 1

4. Create a user interface that is intuitive and appealing to the users.
   Criticality: 3
   Dependencies: None

5. Save a list of prior matches so the matching algorithm does not have to be executed every time the user logs in.
   Criticality: 2
   Dependencies: 1

6. A profile browsing option so each user can browse other potential matches.
   Criticality: 2
   Dependencies: 1

7. After being matched with another user, provide each other with contact information to start the romance.
   Criticality: 2
   Dependencies: None

8. Upload profile picture function.
   Criticality: 2
   Dependencies: None

9. Ability to upload multiple pictures separate from profile pictures.
   Criticality: 1
   Dependencies: None

10. If an account has not been used in 6 months send the user an email and remove account.
3.2 User Interface Requirements

3.2.1 User Interface: Graphical (GUI) or Command-Line (CLI)

The user interface for the software will be a web site. The first time the user visits the web site, they will be able to create an account that will require them to fill out information about themselves and about what kind of person they are looking for. This will be done through the use of labels, text boxes, buttons, and drop down lists.

3.2.2 Application Programming Interface (API)

The application will only be accessible via web interface.

3.2.3 Diagnostics (Error Reporting and Usage Logs)

Users will be able to see the error messages on the screen when errors happen.

3.3 System Requirements

Each user will need a computer that can access the internet.

3.3.1 Hardware Interfaces

This standard web server will be located at the University of Mount Union and will not require any specific hardware interfaces.

3.3.2 Communications Interfaces

The server computer must be able to communicate via the Internet in order for the database to be properly utilized.

3.3.3 Software Interfaces

Users should have up-to-date operating systems. These include, but are not limited to:

- Windows 7 and up.
- OS X 10.7 and up.

3.4 Domain Requirements/Constraints

No domain requirements or constraints exist for this software.

3.5 Non-Functional Requirements

3.5.1 Reliability

The software will function as long as the web server is running and the computer is able to access the Internet.

3.5.2 Colors & Font concepts

There will be appealing color concepts and fonts that will grab the eyes of the user. To make sure these are appealing to users, we will be surveying groups that fit into our audience.
3.5.3 **Availability**
The software shall function so long as the web server is running and able to access the Internet.

3.5.4 **Security**
The security features in place will prevent people without the correct username and password to access an account.
All forms will have a data integrity check to prevent a wrong search term to break the matching algorithm.

3.5.5 **Maintainability**
The software code will be simple enough, with complexity only added where necessary, to allow for current team members and potential other software engineers to perform debugging operations. This will make the code easily maintainable even if the team changes in the future.

3.6 **Logical Database Requirements**
All users and their profile information will be stored in a SQL database. These tables will store information such as name, height, weight, hair color, interests, and so on.
4.1 Choice of Software Life Cycle Model
The Design to Schedule Life Cycle Model will be used throughout the development of this project.

4.2 Justification for Choice of Model
Charming Cupids decided the Design to Schedule Life Cycle Model would be most applicable for this project. This decision was made based on the hard deadline of us graduating in May. This model requires us to implement the highest priority items first, that way when the hard deadline comes the only functionalities that are not implemented would be the least critical functionalities.