2DV609
Project Course in Software Engineering
Assignment D1 - Requirements Document
Group 6

April 16, 2020

Contents

1 Introduction

1.1 Purpose

The purpose of building this software is to provide an easy to understand social platform for users with very little experience with technology. The product will focus on usability and limited, easy to understand, functionality with respect to users special needs, filling a space in the society.

1.2 Scope

The scope of this project is to create a social platform targeted for users with very little experience with technology. The website will need to have a design that is easy to use and understand. The users will be able to add/delete contacts, send messages, send pictures. This messaging service will be free of use. To be able to use the platform, an account will need to be created.

1.3 Stakeholders

This section will list all stakeholders as well as a short description of each stakeholder.

Stakeholders

- **Software Developer:** A software engineer that works with developing, testing and maintaining software.
- **End-User:** An elderly person with a desire to connect with other people sharing the same interests.
- **Examinator:** An experienced software developer that will monitor and evaluate the project.
- **GDPR institute:** An institute that monitors personal data storage and upholds the law of every persons right to their data.
- **Investors:** The buyers of the system with an financial interest.

1.4 Overview

This requirement document is destined for the development of a social platform. This document is separated into 4 sections. Section 1 includes the purpose, scope, stakeholders, overview and definitions. Section 2 contains the initial requirements with systematic check-based analysis and systematic validation. Section 3 provides a general description of the project with product perspective, product functions, general constraints, assumptions and dependencies. Section 4 goes through the specific requirements such as interface, functional, non-functional and database requirements. It also includes the test cases generation and the system attributes.

1.5 Definitions, Acronyms and Abbreviations

This section will help and make sure that anybody who reads this document is aware of the language used in this document.a This will contain any uncommon Definitions, Acronyms and Abbreviations that the normal user would not be familiar with.

- **Modular:** Modular programming is the process of subdividing a computer program into separate sub-programs. It allows for multiple developers to work on different tasks without impacting each other
- **Database:** A collection of data stored on online, that can be access with a request.
- **Query:** A query is a request for information from a database
- **Backups:** A backup is when the latest version of the application is saved to the database.

2 Requirements Process

2.1 Initial Requirements

RC1: The system should be modular (Developer)

RC2: The system should be provide information on how to perform actions (End-user)

RC3: The system should follow naming standards to improve code readability (Examinator, Developer)

RC4: The system should be scalable as the maximum number of users is unknown and could increase in the future (Developer, End-User)

RC5: The system should store user information in a structured way that allows deletion of specific user data (GDPR Institute)

RC6: A request to the database should have a response time under 100ms. (Developer)

RC7: The system should be able to handle at least 1000 requests/min. (Investor)

RC8: A request made by a user should take less than 1 second to complete. (End-User)

RC9: The system should support an account/login system where a user signs up for an account.

RC10: The system should support functionality to enlarge the interface for better visuality.

RC11: All messages between users should be stored in the database.

RC12: A user should be able to delete messages stored in his/hers message history.

RC13: A user should be able to save other users as contacts.

RC14: A user should be able to delete saved contacts.

RC15: The system should prompt for confirmation whenever a user tries to perform an action that is not cannot be reversed.

RC16: A minimum text size of 14pt should be used in the interface.

RC17: A user should be able to send messages to saved contacts.

2.2 Systematic check-based analysis

This section will list all of the initial requirements after the analysis, new information regarding each requirement can include:

- A new description that more accurately capture the requirement
- New requirements split from the original
- Deletion if a requirement does not fulfill an apparent purpose or is not well enough explained
- Added additional stakeholders if they are considered to have an interest in that particular requirement.

Analyzed Requirements

Original ID:	RC1
New ID:	FRC1.1
Stakeholders:	Developer, Investor
Comment:	A more thorough description was needed for clarity of the purpose
The system should be build by using a modular approach so that developers can work on different parts of the system without interrupting each other	

Original ID:	RC2
New ID:	FRC2.1
Stakeholders:	End-User
Comment:	
The system should support functionality to provide more thorough information on how to perform certain actions	

Original ID:	RC3
New ID:	-
Stakeholders:	Examinator, Developer
Comment:	This should not be needed to be listed as a requirement as it should be followed always
DELETED	

Original ID:	RC4
New ID:	NFRC4.1
Stakeholders:	Developer, Investor, End-User
Comment:	Clarified that considering scaling of the system should include both new functional as well as non-functional requirements
The system should be build in a way that enables it to scale with new functionality as well as increased number of users	

Original ID:	RC5
New ID:	FRC5.1
Stakeholders:	End-User, GDPR institute
Comment:	This requirement has been split into two different parts as they clarify different requirements
A users private information should be stored in a way that all trace of it can be removed if the user wants it to.	

Original ID:	RC5
New ID:	FRC5.2
Stakeholders:	End-User, GDPR institute
Comment:	See above
A users private information should be stored in a way that all trace of it can be removed automatically after 90 days time	

Original ID:	RC6
New ID:	NFRC6.1
Stakeholders:	Developer
Comment:	Specifying the client to be a web server
A request made by the web server to the database should have a response time under 100 milliseconds	

Original ID:	RC7
New ID:	NFRC7.1
Stakeholders:	Investors, Developer
Comment:	Added developers as stakeholders as this requirement might determine design/deployment choices
The system should be able to handle at least 1000 requests/min without delay building up in the system	

Original ID:	RC8
New ID:	-
Stakeholders:	End-User, Developer
Comment:	This requirement is too ambiguous to be used
DELETED	

Original ID:	RC9
New ID:	FRC9.1
Stakeholders:	Developer
Comment:	Added a more thorough description of how the login system should work
The system should use an account/login functionality which requires a user to be logged in to access the systems other functions	

Original ID:	RC10
New ID:	FRC10.1
Stakeholders:	End-User
Comment:	Changed the description to specify that it should be limited to textual contents as well as it should be an action performed by a user
A user should be able to perform a single action in able to enlarge the textual content	

Original ID:	RC11
New ID:	FRC11.1
Stakeholders:	End-User
Comment:	
A user should be able to send a textual message to another user and the message should visual to both users	

Original ID:	RC11
New ID:	FRC11.2
Stakeholders:	End-User
Comment:	
All textual messages between users should be stored in the database together with sender/recipient information	

Original ID:	RC12
New ID:	FRC12.1
Stakeholders:	End-User
Comment:	Specified that a deletion action must be performed in order to delete a message
A user should be able to delete a message from the database by performing a deletion action on a selected message	

Original ID:	RC13
New ID:	FRC13.1
Stakeholders:	End-User
Comment:	Added a more thorough description
A user should be able to save another users profile as a contact by performing an add-contact-action on a selected user profile	

Original ID:	RC13
New ID:	FRC13.2
Stakeholders:	End-User
Comment:	Added a more thorough description
A users should be able to remove a saved contact by performing a remove-contact-action on a selected user profile	

Original ID:	RC15
New ID:	FRC15.1
Stakeholders:	End-User
Comment:	Unchanged
The system should prompt for confirmation whenever a user tries to perform an action that is not cannot be reversed	

2.3 Systematic Validation

In this section the work of validating the requirements will be presented as a table where each requirement gets marks if they fulfill a point in the checklist.

In case a requirements is deemed not to be valid a rationale will be given in the end of this section.

The checklist includes:

[C1] **Completeness:** There is not any requirement missing or missing information from the individual requirement description

[C2] **Consistency:** The description does not include contradictions

[C3] **Comprehensibility:** The requirement is written in a way that is understandable

[C4] **Ambiguity:** There are not multiple ways of interpret the requirement

[C5] **Structure:** This requirement is organized so that it is grouped with other relatable requirements

[C6] **Traceability:** This requirement can be be traced to related requirements

[C7] **Standards:** The individual requirement conform to defined standards

ID	C1	C2	C3	C4	C5	C6	C7	Valid
FRC1.1	X	✓	X	✓	✓	✓	✓	X
FRC2.1	✓	✓	✓	✓	✓	✓	✓	✓
NFRC4.1	✓	✓	✓	✓	✓	✓	✓	✓
FRC5.1	✓	✓	✓	✓	✓	✓	✓	✓
FRC5.2	✓	✓	✓	✓	✓	✓	✓	✓
NFRC6.1	✓	✓	✓	✓	✓	✓	✓	✓
NFRC7.1	✓	✓	✓	✓	✓	✓	✓	✓
FRC9.1	✓	✓	✓	✓	✓	✓	✓	✓
FRC10.1	✓	✓	✓	✓	✓	✓	✓	✓
FRC11.1	✓	✓	✓	✓	✓	✓	✓	✓
FRC11.2								
FRC12.1								
FRC13.1								
FRC13.2								
FRC15.1								

Table 1: Requirements Validation Table

Invalid Requirements

- **FRC1.1:** The structure of the project code does not fit as a requirement and writing a test case for this requirement would be hard.

3 General Description

3.1 Product perspective

BrandName is a new online messaging service that allows the user to connect and talk to other users, using a simple interface that showcases chats in text bubbles complete with timestamps and notifications when the recipient reads the message. Our system aims to develop a smooth and enjoyable experience for the user using simple but clear interface, where every action happens in one main screen with clear buttons, column sand icons that will allow the user to have a good and intuitive feeling when using it for the first time. BrandName does not work independently since it uses internal servers to send and receive messages on both end devices.

3.2 Product functions

BrandName software should support computerized network over the internet, providing the user with simple but key features. The main feature of our application is the instant messaging across users including text messages, pictures and audios. Aside from the main feature, our application also has some secondary features such as create group chats with any contact from the friend list, change profile picture, edit email address and delete account. BrandName uses the customer email address and an ID number (user1@mail.com#1) to identify the user once the member is validated by the system. This software will collect information about a simple account transaction (e.g., sent messages, received messages, media transfers, and network usage) while it communicates with the internal severs for message processing and delivers the message to the receiver in time to receive their reply.

3.3 General constraints

While developing this instant messaging application some constraints should be noted:

1. Create an account by entering and verifying the email address.
2. Access large amounts of data with a low internet speed.
3. Handling different types of video and audio formats in the database
4. Server capacity
5. Account security

3.4 Assumptions and dependencies

In order to make a fully operational application and to insure that everything runs accordingly, the following criterias are needed:

1. The user has a viable and stable internet connection.
2. The user has a device with access to a web browser.
3. The server capacity is large enough to host multiple clients.
4. The hardware used never fails.
5. No packets are lost between the server and client.

4 Specific Requirements

4.1 Interface requirements

4.2 Functional requirements

4.2.1 FR1 - User Registration

Original ID: **New ID:**	RC9 FRC9.1
Description	The system must be able to register any new user through a username, valid email address and a valid password otherwise the user account will not be validated by the system and will not be able to create the desired account.
Input	Username/ Email and password using a keyboard.
Process	Validate email address.
Output	System displays an *"Account successfully created"* message.

4.2.2 FRC9.2 - User Login

Original ID: **New ID:**	RC9 FRC9.2
Description	The system must be able to log in any already registered member.
Input	Username/ Email and password using a keyboard.
Process	Store the information in the database and check if the member already exists, send a verification code so the login can be executed with no problems.
Output	The system displays a welcome/login message and redirects the user to the homepage.

4.2.3 FRC13.1 - Friend Request

Original ID: **New ID:**	RC13 FRC13.1
Description	The system must be able to locate and send a friend request to a specific member already registered in the database.
Input	Username or Email from the desired contact.
Process	Systems checks with the database for any members with the correspondent input data and sends a friend request to the desired one.
Output	A friend request is sent by the system.

4.2.4 FRC13.2 - Add new contact

Original ID: **New ID:**	RC13 FRC13.2
Description	The system must be able to find and add a new contact to the user s friend list. The system will only be able to find and add the contacts if the receiver accepts the friend request.
Input	Friend request.
Process	Systems checks with database for any change in the friends request state.
Output	The system displays the user friends list displaying all the contacts.

4.2.5 FRC17.1 - Send message

Original ID: **New ID:**	RC17 FRC17.1
Description	The system must be able to send an instant message to any contact on the user's friend/contact list. The system must notify the sender when the message is successfully delivered to the receiver by displaying a tick icon next to the message sent.
Input	Message.
Process	System sends an instant message to the other user.
Output	Displays a tick icon under the message.

4.2.6 FRC17.2 - Send Multimedia Attachments

Original ID: **New ID:**	RC17 FRC17.2
Description	The system must support Multimedia Attachments in their messages so the user can send audio, video and image as attachments.
Input	File attached.
Process	System sends an instant message to the other user.
Output	Displays a tick icon under the message.

4.2.7 FRC17.3 - Message Status

Original ID:	RC17
New ID:	FRC17.3
Description	The system must be able to get information on the message state to provide all that data to the user when sending a message to someone. If the receiver reads the message, 2 tick icons must appear under the message.
Input	Message.
Process	System sends an instant message to the other user and checks if the receiver already read it.
Output	Displays two tick icons under the message.

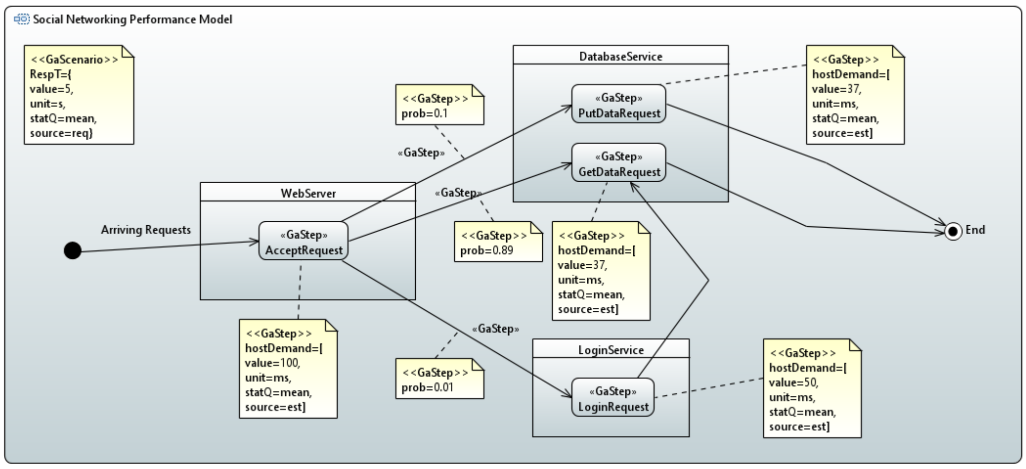
4.3 Non-functional requirements

4.3.1 Performance

- Performance is the time at which a content is delivered to users and how the system response to it.
- Processing time - Communications with the server which require processing such as login should take less than five seconds.
- Response Time - With a reasonable common internet connection speeds, the server should respond to client request in less than a second.
- Querying Time - Querying the database should take less than a second.

4.3.2 Performance Model (MARTE)

A performance model was done with estimated numbers for path probabilities and host demands. As all of the deployment options has not been decided the current model is the most probable interaction between the system and services.



Social Networking Performance Model
<<GaScenario>> RespT={ value=5, unit=s, statQ=mean, source=req}
DatabaseService
«GaStep» PutDataRequest
«GaStep» GetDataRequest
<<GaStep>> hostDemand=[value=37, unit=ms, statQ=mean, source=est]
<<GaStep>> prob=0.1
«GaStep»
WebServer
«GaStep»
Arriving Requests
«GaStep» AcceptRequest
<<GaStep>> prob=0.89
<<GaStep>> hostDemand=[value=37, unit=ms, statQ=mean, source=est]
End
<<GaStep>> hostDemand=[value=100, unit=ms, statQ=mean, source=est]
«GaStep»
<<GaStep>> prob=0.01
LoginService
«GaStep» LoginRequest
<<GaStep>> hostDemand=[value=50, unit=ms, statQ=mean, source=est]

Figure 1: MARTE Performance Model

4.3.3 Capacity and scalability

Capacity is the amount of resources made available to the system, and scalability is the ability to make use of the system of the available resources.

- **Storage** The system must store data effectively and must prepare for the time remaining until all available storage is filled up.
- **Growth requirement** As the system gets used more often, the storage available will need to increase.
- **Demand** The system should be able to handle 1000 queries per minute, the the ability to increase with large demand.

4.3.4 Availability

Availability is the ratio of time the system is online.

- Hours of operation - The system should be usable during the hours it is mostly used. Any maintenance where the system needs to be shut down should be done at odd times where it is not been used by many users.
- **Locations of operation** The geographic location of the server should not curb the availability of the system.

4.3.5 Maintainability

Maintainability describes how well the system can be kept functional and how well it can be changed.

- Architectural requirements - The design patterns should be used in the construction of the base architecture.
- Coding requirements - The code should be build standardized, common coding style should be applied.

4.3.6 Recovery

Recovery is the ability for a system to prepare and respond to failure.

- Backups - The system should be responsible to taking backups of data, such that it may be restored to a working state.
- Backup frequency - The system should backup data very frequently (Example, Every fifteen minutes) to avoid any data loss.
- Backup time - The system should backup in a short period of time (Example, every one minute) with minimal disruption.
- Recovery time - IN event of a disaster, the latest backup should be immediately restored, such that the system is offline for less than one hour.

4.3.7 Security and privacy

Due to sensitive information being contained within the database, the system should be facilitate security and privacy.

- **Authentication** Users must be logged in order to see post and people,s profile.
- **Authorisation** Users must be friends with the other user in order to see what he post.
- **Confining of sensitive information** Password must not be stored within the system or revealed to users.
- **Privacy** Users must be able to choose how who can see their profile and posts.

4.3.8 Mobility

Mobility describes the suitability of the system for platforms.

- Mobile devices - The system should be operational on wide spectrum of operating condition, such as different screen sizes, performance and internet connection speed.
- Input devices - The system should make effective use of input devices on the mobile device. For example, the camera should be used for photo and filming.

4.4 Test case generation

This section will propose a test case for each requirement. This is mainly to validate the requirement by proving that it can be tested.

Test Cases

Requirement ID:	RC1.1
Test Case ID:	TC1.1
Test Steps	
1.	Inspect the code structure
Result:	The functionality of each package should be accessed by a facade class

Requirement ID:	RC2.1
Test Case ID:	TC2.1
Test Steps	
1.	The user activates the help menu
2.	The system opens the interface for the help menu
3.	The user selects what action he/she wants more information about
Result:	The system provides more information about the selected action

Requirement ID:	RC4.1
Test Case ID:	TC4.1
Test Steps	
1.	Simulate usage of the system with an increasing number of users
Result:	The system should be able to handle more users than expected

Requirement ID:	RC5.1
Test Case ID:	TC5.1
Test Steps	
1.	The user performs the action to delete the account
2.	The system prompts the user for confirmation
3.	The user confirms
Result:	The system responds that the account has been deleted along with all account specific information

Requirement ID:	RC5.2
Test Case ID:	TC5.2
Test Steps	
1.	Store test data in the database
Result:	After 90 days the system should have deleted the test data

Requirement ID:	RC6.1
Test Case ID:	TC6.1
Test Steps	
1.	Use a web server performance testing software and monitor the results
2.	test
Result:	Response time between the web server and database should not be above 100 milliseconds

Requirement ID:	RC7.1
Test Case ID:	TC7.1
Test Steps	
1.	Use a web server performance testing software to make 1000 requests/min and monitor the results
Result:	The delay for each request to complete should be within reasonable numbers, and the delay should not grow exponentially

Requirement ID:	RC9.1
Test Case ID:	TC9.1
Test Steps	
1.	Try to access other functionality than the login functionality without being logged in.
Result:	The system should deny access to all of the functions

Requirement ID:	RC10.1
Test Case ID:	TC10.1
Test Steps	
1.	Perform an action meant to enlarge all textual content
Result:	The system should have enlarged all textual content of the interface

Requirement ID:	RC11.1
Test Case ID:	TC11.1
Test Steps	
1.	Send a textual message from one user to another
Result:	The textual message should visible to both the sender and receiver

Requirement ID:	RC11.2
Test Case ID:	TC11.2
Test Steps	
1.	Send a textual message from one user to another
Result:	The textual message should be located in the users folder in the database

Requirement ID:	RC12.1
Test Case ID:	TC12.1
Test Steps	
1.	Send a textual message from one user to another user
2.	The system should notify that the message has been send and the message should be visible
3.	Perform the delete-message-action
Result:	The system should notify the user that the message has been deleted and the message should be removed from the database

Requirement ID:	RC13.1
Test Case ID:	TC13.1
Test Steps	
1.	When logged in, select another user and perform the action to add-contact
Result:	The system should confirm that the user has been added as a contact and the user should be located under contacts in the database

Requirement ID:	RC13.2
Test Case ID:	TC13.2
Test Steps	
1.	test
Result:	results

4.5 Database requirements

This section will list requirements related to the database.

- [**FRC5.2**] A users private information should be stored in a way that all trace of it can be removed automatically after 90 days time
- [**NFRC6.1**] A request made by the web server to the database should have a response time under 100 milliseconds
- [**NFRC7.1**] The system should be able to handle at least 1000 requests/min without delay building up in the system