SRS Requirements Document

Version 1.3

In Preparation for:

Computer Science 4770

Authored by:

Kristan Hart
Keir Strickland-Murphy
Diego Zuluaga
Meishang Chen
Aizaz Iqbal

Table of Contents

1 Introduction

1.1 Purpose

The purpose of this SRS is to document the development of the MUN Student's Social Network (MUNSSN) for Computer Science 4770 (Team Project) by Team C. The intended features for this project can be found in the MUNSSN Project Description[1].

1.2 Scope

The scope of this project includes the five members of Team C, Diego Zuluaga, Meishang Chen, Kristan Hart, Keir Strickland-Murphy, and Aizaz Iqbal. The features of the project will be implemented using Node.js, MongoDB for the databasing, and GitHub for version handling and distribution. The main goal of this project is to provide a social network for students in Memorial University. Students should be able to sign up, have a timeline, create groups, post lost & found items and create polls.

1.3 Definitions & Abbreviations

- MUNSSN - Memorial University of Newfoundland Student's Social Network, the name of the project being developed.
- User - Someone who has not yet signed up to the MUNSSN
- Member - Someone who has signed up for the MUNSSN

1.4 References

1. MUNSSN Project Description
2. SRS Template

2 Overall Description

2.1 Product Perspective

The MUNSSN is a standalone product being created for Memorial University. It will serve as a social network among students of the class with expected functionalities from a standard project of this type.

2.2 Product Functions

- Users will be able to sign up to become members of the social network
    - User will enter profile information including a profile picture and chooses a username and password
    - Username will be user's MUN email address, and they will have to verify their email address by replying to the confirmation email
- Members can send friend requests to other members, and can accept or decline requests from other members
- The network will show a list of recommended friends
- Members have a timeline where they can post content
- Members can set visibility for their published post, limiting who can view them
    - Should be able to do any of the following, and can modify posts after publishing
        - Only I can see my published posts
        - Everyone can see my published posts
        - Only my friends can see my published posts
        - Only a specified list of friends can see my published posts
- Members should be able to define who can post on their timeline
    - Only them
    - Everyone
    - Friends
- Members can comment on posts based on visibility
    - Comments can be replied to
    - Comments can be edited
- Members can create public or private study groups
    - Member who creates the group becomes the owner
    - Owner can invite other members
    - Owner specifies if other members can invite

    - If group is public members can join without being invited
    - If group is private members can only join after being approved by the owner
    - In group members can share course contents
- Lost and found functionality where members can post pictures and locations of lost items that show up on a map, or contact the member who posted to the lost and found to coordinate pickup
- Members have schedules that can be viewed by other members
- Members have a section to upload their resume which can be viewed by their friends
- Members should be able to create polls and then choose the audience to view/ vote on the poll. If the audience is students taking the course, then they will be the only ones able to view and vote on the poll, however if the poll is public, any member of the social network will be able to view and vote.

2.3 User Classes and Characteristics

- Users which are not yet members and have limited access to the MUNSSN, only being able to view the main page and the signup page.
- Students will have full access to their own profile. Their ability to view other members and friends will be limited to those user's profiles.
- Teachers and Teaching Assistants will have the same rights as other members plus being able to modify the courses information.
- Super User account used for testing and modifying the network

2.4 Operating Environments

- The product will be implemented using Node.js, MongoDB, and version control will be handled using GitHub. It will be accessed through any internet browser, and hosted on the Memorial University servers.

2.5 Design and Implementation Constraints

- The project must be completed using services already implemented on the Memorial servers, and will have a time constraint enforced by the due dates for various steps in the development process. These dates come from the course requirements. The use of Node.js, MongoDB, and GitHub are not merely selected for use in the project, but required by these same course requirements.

3 Functional Descriptions

3.1 Signup

3.1.1

Introduction:The first steps in using this site will be the signup process in which the new user will enter a series of personal information and decide on a username and password for the account(discussed in 3.2 and 3.3).

3.1.2

Input: string format as this is all simple data to be stored.

3.1.3

Processing:the data is only meant to be stored for later display purposes.

3.1.4

Output:confirmation email found in section 3.4 and 3.5.

3.1.5

Error handling: checking for invalid inputs i.e. numbers in the name, special characters in the name, etc.

3.2 Email Requirements

3.2.1

Introduction:Users will be required to enter a memorial university email as their username in this site as it is only meant for MUN students.

3.2.2

Input: single string which will be the MUN email address.

3.2.3

Processing: storage of the email address.

3.2.4

Output: confirmation email found in section 3.4 and 3.5.

3.2.5

Error handling :invalid emails addresses, including non-existant addresses and email addresses which are not MUN's. We also will be checking against already registered users.

3.3 Password Creation

3.3.1

Introduction:At time of account creation users will be required to make a password for the account that must abide by the following rules

I) Must be longer than 8 characters

II)Must include a some capital letters

III)Must include some number or special characters

3.3.2

Input:string.

3.3.3

Processing:The password will be stored in the new account. Its will be hashed and protected using bitcript

3.3.4

Output: N/A

3.3.5

Error handling:checking the password against the requirements to ensure legal input

3.4 Sending Confirmation Email

3.4.1

Introduction:A confirmation email will be sent to any user who signs up to ensure a valid sign up and activate the account.

3.4.2

Input:address the for the email the user supplied.

3.4.3

Processing: The email is auto-generated and given the input address.

3.4.4

Output:The email is sent to the given address.

3.4.5

Error handling: checking for failed delivery.

3.5 Confirmation Email Response

3.5.1

Introduction:Once the confirmation email is received the user must click the activation link to activate the account.

3.5.2

Input: confirmation link that the user needs to click.

3.5.3
Processing:The account will be marked as active.
3.5.4
Output: N/A
3.5.5
Error handling: checks to ensure no confirmation fails.

3.6 Friends List

3.6.1
Introduction:Members will start with an empty friends list and will be able to add to it at any time.
3.6.2
Input: reference to the new friend to be added.
3.6.3
Processing: The given reference to a friend will be checked against the user database and the the request will be generated
3.6.4
Output:Request will be sent to the person who the user wishes to be friends which will then sit idle until the second user either accepts or denies it.
3.6.5
Error handling:failed sends and if the user the request is being sent to does not exist i.e. input errors for the name.

3.7 Suggested Friends

3.7.1
Introduction: A list of suggested friends will be available to all users based on friends of friends, same courses, programs etc.
3.7.2
Input: Criteria such as friends, courses and programs will be input from the profile.
3.7.3
Processing: A list of potential matches will get generated based on given criteria.
3.7.4
Output: suggested friends section of the users page.
3.7.5
Error handling :failed database searches for users based on invalid inputs

3.8 User Timeline

3.8.1

Introduction: Each member will have a timeline on their page where they will be able to post video, text and pictures.

3.8.2

Inputs will include video files, string inputs and img files.

3.8.3

Processing will include moving the inputted data into storage in the system and attaching a date to the post.

3.8.4

Output will include displaying the text and picture on the page in chronological order

3.8.5

Error handling will include checking for failed picture uploads and failed posts.

3.9 Timeline Visibility

3.9.1

Introduction:Users will be able to set visibility on any given post at any given time with the following conditions :

i. Only I can see my published posts.

ii. Everyone (considering users) can see my published posts.

iii. Only my friends can see my published posts.

iv. Only a specified list of friends can see my published posts.

3.9.2

The input will be a list of people allowed to see the post

3.9.3

The post will be marked as visible to the given list.

3.9.4

Output will be confirmation the visibility has been set

3.9.5

Error handling will include failed requests to set visibility.

3.10 Timeline Posting Permissions

3.10.1

Users will be able to set who is allowed to post to their timeline with the following conditions:

I. Only I can post on my timeline.

II. Everyone (not including users) can post on my timeline.

III. Only my friends can post on my timeline.

3.10.2

Input will be some variable to define who is allowed to post on the timeline.

3.10.3

Timeline will be marked such that only the defined type of user is allowed post on it.

3.10.4

Output will be confirmation the permissions have been set

3.10.5

Error handling will include checks for failed attempts set the permission.

3.11 Posting Comments

3.11.1

Introduction:Members can put comments on each others' posts based on the visibility of the post.

3.11.2

Input: String, text and emoticons

3.11.3

Processing: System will store the text into database

3.11.4

Output: Text will be placed below the post

3.11.5

Error Handling: Query injection and internet connection time out

3.12 Reply Comments

3.12.1

introduction : Comments can be replied to at all times

3.12.2

Input: String /Text

3.12.3

Processing: System will convert the text into web element and store it into database

3.12.4

Output: Text will places below the first comment

3.12.5

Error Handling: Query injection and internet connection time out

3.13 Edit Comments

3.13.1

introduction:Comments can be edited at all times and a history of the previous comments should be shown to a user when they click on see previous versions.

3.13.2

Input: String/ Text

3.13.3

Processing: system will replace the current comments to the latest edit

3.13.4

Output: New comments

3.13.5

Error Handling: Query injection and internet connection time out

3.14 Create Group

3.14.1

Introduction: Each member can create either a public or private study group .

3.14.2

Input:Group name, group type(public, private),group description

3.14.3

Processing: System will take all information and store into database and user who creates the group becomes the group owner.

3.14.4

Output: Group page will be created

3.14.5

Error Handling:if the group name had been taken and internet connection time out

3.15 Group Invite

3.15.1

Introduction: Group owner can specify if other members can invite others to the group or not.(invite permission)

If the group is public, other users can join the group instantly.

If it is private, the users can join the group only after the owner accepts the request, or by invitation from the group owner, or other members, if they are allowed to do so.

3.15.2

Input: selected Friends

3.15.3

Processing: system will take the selected person and send a request to that member

3.15.4

Output:group invitation or permission

3.15.5

Error Handling: Internet disconnection

3.16 Post on group

3.16.1

Introduction: Each member can post on timeline

3.16.2

Input: String/ Text, picture

3.16.3

Processing: system will take the string and picture into web elements and store into database

3.16.4

Output: Webview of the post

3.16.5

Error Handling: internet disconnect and query injection

3.17 Share course contents

3.16.1

Introduction: members can share course contents(note, reading material,file) together

3.16.2

Input: file, document, pdf

3.16.3

Processing: file will store into database and create link for download

3.16.4

Output: download link

3.16.5

Error Handling: The file is too large, and uploads will be limited to pdf files

3.18 Lost and found section

3.18.1

Introduction:The network will have a lost and found section and member can access it anytime

3.18.2

input : item description, photo, map location

3.18.3

Processing: system will convert the input to web elements

3.18.4

Output: Web post

3.18.5

Error Handling: file too large, location not identify

3.19 Message the original poster

3.19.1

Introduction:users have the ability to message the original poster

3.19.2:

Input: N/A

3.19.3

Processing: member click on the poster, if the poster is a friend, they will have access to their phone number, if not, the available information would be the poster's email address)

3.19.4

Output: poster's phone number/ email

3.19.5

Error handling: no relative error

3.20 Upload Schedule

3.20.1

Introduction: Members can add schedule to the their profile , initialized by entering the date and hours of classes

3.20.2

Input: course name, date, time slot

3.20.3

Processing: system will take the information and convert into schedule view

3.20.4

Output: Calender View of the schedule

3.20.5

Error Handling: duplicate course, conflict time slot

3.21 Upload Resume

3.21.1

Introduction: Members can upload resume to their profile, which will be can be viewed by all friends

3.21.2

Input: resume pdf. doc file

3.21.3

Processing: system will take the file and display it on their profile

3.21.4

Output: webview of the file

3.21.5

Error Handling: File too large, only pdf files.

3.22 Create Poll

3.22.1

Introduction:Members can create polls about courses

- Only friends in the same course can respond to the polls posted
- Poll owner can modify and delete poll

3.22.2

Input: course name, option for course

3.22.3

Processing: system will convert the information to poll view

3.22.4

Output: poll view

3.22.5

Error Handling: course does not exist

4 External Interface Requirements

4.1 User Interfaces

Initially the member of the mobile application will see the login screen in order to gain access to the website. If they have not yet registered, they will be redirected to the sign in page. users will not be able to access the website.

Members will have a profile page which they can edit their information (email address, address, personal info, etc.) post/ view their timeline, check their schedule and start a poll/ post on a poll for a course they are signed into. Graphical representations of these pages are included in figure 1, 2 and 3 below.

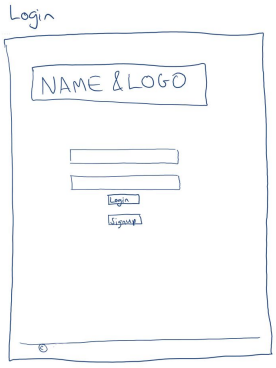


Figure 1: Login Screen

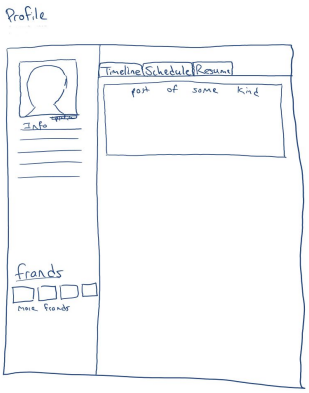


Figure 2: Profile Page

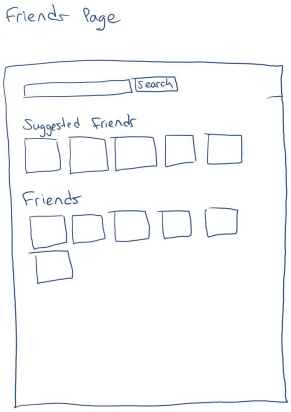


Figure 3: Friend Page

Member will be able to join a group. Groups can be searched, but recommended groups will be on a list on the user group page (figure 4). Group pages will have their own design and function (Figure 5). Users accepted to the group will be able to post on the front page of the group page. The Lost and Found page will allow users to search for lost items, and to post lost items with their location. Anyone can post and search on this page (figure 6).

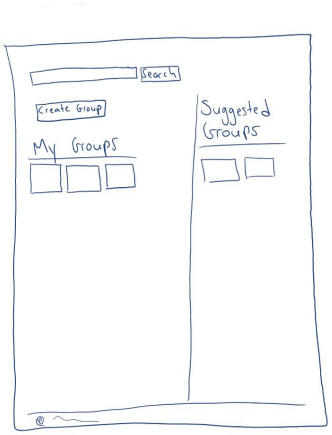


Figure 4: User Group Page

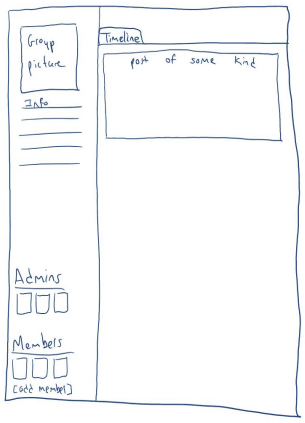


Figure5: Group Page

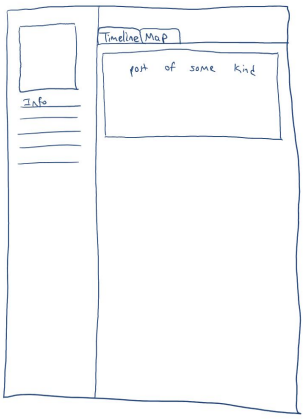


Figure 6: Lost/Found Page

4.2 Hardware Interfaces

While the web application is not itself a hardware based system, it will be interacting with either a wireless device or a desktop device. The only other hardware requirements for the software is the ability to gain a GPS signal and have access to the internet.

4.2.1 Mobile

Name of Product: Wireless device, supporting HTML5.

Description of purpose: To gain access to the social network site, using touch and text input

Source of input or destination of output: Input from touch commands or text, the destination of the output is through the server and into the mongo database

Valid range, accuracy and/or tolerance: Dependant upon the wifi signal of the wireless device

Relationships to other inputs/outputs: Data entered in the front end side of the application will impact upon both the server and the database.

Screen formats/organization: No restrictions on screen size

Window formats/organization: Vertical and horizontal orientation.

Data formats: Data will be sent to the server as text, and the login as encrypted text.

Command formats: No command prompts needed, all is done via the GUI.

4.2.2 Desktop

Name of Product: Desktop device, any OS, web browser supporting HTML5.

Description of purpose: To gain access to the social network site, using mouse clicks and text input

Source of input or destination of output: Input from mouse commands or text, the destination of the output is through the server and into the mongo database

Valid range, accuracy and/or tolerance: Dependant upon the wifi signal of the device or the internet connection speed if connected via ethernet.

Relationships to other inputs/outputs: Data entered in the front end side of the application will impact upon both the server and the database.

Screen formats/organization: No restrictions on screen size

Window formats/organization: Horizontal.

Data formats: Data will be sent to the server as text, and the login as encrypted text.

Command formats: No command prompts needed, all is done via the GUI.

4.3 Software Interfaces

The web application requires a variety of interconnected software in order to function. For the purposes of our application we are using a variation of the MEAN stack, (Mongo, Express, AngularJS and NodeJS) as well as the front end HTML and CSS. The front end and the javascript need to communicate to the server through express and Nodejs and in turn, these need to communicate to the Mongo database in order to store the information from the website. Since we are using HTML5, the application will work with either an Android, Windows, Blackberry or IOS environment.

4.3.1 Database software

Name of Product: MongoDB database

Description of purpose: Store the data of the website as BSON data.

Source of input or destination of output: Input is from the website, using EJS (an express template), through nodeJS into the database.

Valid range, accuracy and/or tolerance: Range is the defined by the device's wireless ability.

Timing: Timing should be in the millisecond range, including refreshing the server after crash, as well as any data lookup.

Relationships to other inputs/outputs: Data entered will be entered on the site through HTML5 and EJS and which connects to the server through NodeJS and then into the MongoDB.

Screen formats/organization: N/A

Window formats/organization: N/A

Data formats: BSON

Command formats: Mongo based commands on the terminal.

4.3.2 Server-side Software

Name of Product: NodeJS

Description of purpose: Event-driven I/O server-side JavaScript environment, for communicating both the front end to the server and then from the server to the backend (in this case MongoDB).

Source of input or destination of output: Input is from the website and the destination is into the MongoDB

Valid range, accuracy and/or tolerance: Accuracy is determined by the setup environment.

Timing: If connection is present, server interactions should be instantaneous

Relationships to other inputs/outputs: Data entered from the front end will be placed into the database

Screen formats/organization: N/A

Window formats/organization: N/A

Data formats: Data is text based, and javascript data (JSON) through to be stored into the mongoDB as BSON data.

Command formats: Terminal commands depending on the operating system used.

4.3.3 Scripting Software

Name of Product: Javascript

Description of purpose: To give added functionality to web applications in order to dynamically display information.

Source of input or destination of output: Input is from the website, and accessed by a javascript file through the html.

Relationships to other inputs/outputs: Data entered from the front end will be placed into the database

Screen formats/organization: N/A

Window formats/organization:N/A

Data formats: JS data, JSON, BSON, html, txt

Command formats: Terminal commands depending on the operating system used.

4.3.4 Front-End Software

Name of Product: HTML/CSS

Description of purpose: To show all of the information on the web page.

Source of input or destination of output: Input is from the user and the output from the web page goes to the server.

Valid range, accuracy and/or tolerance: N/A

Units of measure: Milliseconds, the webpage should work as fast as the internet connection allows

Timing: Instantaneous.

Relationships to other inputs/outputs: Input from the web page will interact with the javascript, ejs and nodejs

Screen formats/organization: No restrictions on screen size.

Window formats/organization: Any.

Data formats: html, css, ejs, js, txt

Command formats: Terminal commands depending on the operating system used.

4.4 Communications Interfaces

The application uses Memorial University emails to authenticate users, but does not have an email server on its own. Both the user (client) and the admin will be using http/https protocol while connected to the internet. The application will access the wifi via the local server, through NodeJS. The web application should work with any fully updated modern web browser, however there is some concern with the google maps api and its functionality with chrome web browser.

5 System Features (Use Cases)

This section describes the functional requirements of the application and the features it provides. The ordering of the system features should be the general order of a new user encountering the site for the first time. System features are used to help with future iterations of the project.

5.1 Initial signup

5.1.1 Name:

Initial signup

5.1.2 Goal:

Create new account

5.1.3 input:

Strings for username, password and other assorted personal information for the account.

5.1.4 Output:

Confirmation email sent to the given email address

5.1.5 Steps:

1. Enter valid email into given text field
2. Enter valid password(meeting preset requirements) into given text field
3. Enter assorted personal information into given text fields
4. Optional: Upload profile picture
5. User must click link in confirmation email to activate account

5.1.6 postcondition:

Account created and active able to use all functionalities of the site.

5.2 Log In

5.2.1 Name:

Log In

5.2.3 Goal:

To successfully enter the username and password and gain access to the account.

5.2.4 Input:

Two string in two separate input spaces, where one is the email associated with the account and the other is the password for the account.

5.2.5 Error Handling:

Each text field has its own error handling where it notifies the user that the given field is incorrect.

5.2.6 Step:

1. User enters a valid email address that has an associated account
2. User enters a valid password for the account that is associated with the email.
3. User Is brought to the profile page of the account they have logged into

Alt Case 1:

User enters an invalid email and is given a error message and must enter a new one.

Alt Cae 2:

User enters an incorrect password for the given account and must enter another one

5.2.7 Postcondition:

User is now on their profile page and is able to access the sites features.

5.3 Adding Friends

5.3.1 Name:

Adding friends

5.3.2 Goal:

Add a new friend to the friends list

5.3.3 input:

Strings for username of person being requested or clicking on a button in the GUI to add a new person

5.3.4 Output:

Notification of freind request for the person being added will be sent to their page

5.3.5 Steps:

1. User identifies another user to be added though suggested friends, or other search
2. User sends request
3. Other user must accept or reject request

5.3.6 postcondition:

New friend may or may not be added at a time of the person being requested choosing.

5.4 Posting to Timeline

5.4.1 Name:

Posting to timeline

5.4.2 Goal:

To post either text or pictures and set visibility and permissions of the timeline

5.4.3 input:

Text block and/or picture file to the timeline section of the profile

5.4.4 Steps:

1. User enters some text to timeline
2. Optional: User uploads a picture to be posted
3. User sets visibility of the post (everyone,friends,self only)
4. User sets permission of who can post on timeline and individual post(everyone,friends, specified friends, self)

Alt Case 1:

User adds some combination of text and pictures to another user's timeline and must abide by the permissions of that timeline.

5.4.5 Postcondition:

A new post has appeared on the users timeline with the permission the user wanted.

5.5 Posting Comments:

5.5.1 Name:

Put comments on each others' posts

5.5.2 Goal:

User post a comments to a post

5.5.3 Input:

String/text

5.5.4 Output:

Comments will display below the post

5.5.5 Main Scenario:

member A see a post from a friend, so he wants to comments on his friend's post

5.5.6 Pre-condition:

Members have visibility of the post

Members have right to comment on the post

5.5.7 Step:

1. Member A view the posting from other friends
2. He clicks on the textbox below the post
3. Enter the comment
4. Click reply
5. Comment shows below the post

5.5.8 Post-condition:

Comments can be replied to at all times

Comments can be edited at all times.

5.5.9 Exceptional Scenario 1

When the user tries to post their comment, but the internet is not connected at that moment, this will cause failure to post the comment

5.5.10 Exceptional Scenario 2

User try to post their comment as query code, this will cause failure to post the comment because the system will detect as query injection.

5.6 Replying to a Comment:

5.6.1 Name:

Reply comments

5.6.2 Input:

text/string

5.6.3 Output:

Replied comment show below original comments

5.6.4 Main scenario:

member A saw member B comments on his post, so user A want to reply member B's comments

5.6.5 Precondition:

User have visibility of the post and comments

5.6.6 Step:

1.user click reply on original comments
2. User enter comment into textbox
3. Click enter to reply
4. Comment shows below the original comment

5.6.7 Post-condition:

Other user can reply to same comments as well

5.6.8 Exceptional Scenario 1:

When user try to post their comment, but internet connect was disconnected at that moment, this will cause failure to post the comment

5.6.9 Exceptional Scenario 2:

User try to post their comment as query code, this will cause failure to post the comment because the system will detect as query injection.

5.7 Edit comment

5.7.1 Name:

Edit comment

5.7.2 Input:

String /text

5.7.3 Output:

New comment and old comment

5.7.4 Main scenario:

Member wants to change his reply from "hello world" to hello class

5.7.5 Pre-condition:

User already post or replied a comment

5.7.6 Step:

1. User click on edit
2. Enter the new comment " hello class"
3. Click enter to confirm endit

5.7.7 Post-condition:

A history of the previous comments should be shown to a user when they click on see previous versions.

5.7.8 Exceptional Scenario 1:

When user try to post their comment, but internet connect was disconnect at that moment, this will cause failure to post the comment

5.7.9 Exceptional Scenario 2:

User try to post their comment as query code, this will cause failure to post the comment because the system will detect as query injection.

5.8 Create/Manage Group

5.8.1 Name:

Create study Group

5.8.2 Input:

Group name, group type(public/private), group description

5.8.3 Output:

New Group page will be create for the group

5.8.4 Pre-condition:

User must be login to the network and account must be activate

5.8.5 Main scenario:

Member A want to create a group name , group type, group group description

Step

1. User navigate to create group page
2. Enter name, type, description
3. Click enter to confirm
4. Group page will automatically create

5.8.6 Post-condition:

- User who creates the group becomes the group owner.
- The owner can invite other members to join the group.
- Group owner can specify if other members can invite others to the group or not.
- If the group is public, other users can join the group instantly.
- If it is private, the users can join the group only after the owner accepts the request, or by invitation from the group owner, or other members, if they are allowed to do so.

5.8.7 Exceptional Scenario 1:

When user try to create the group, but internet connect was disconnect at that moment, this will cause failure to post the comment

5.8.8 Exceptional Scenario 2:

User try to enter the information of the group as query code, this will cause failure to post the comment , Because the system will detect as query injection.

5.9 Lost and Found Item

5.9.1 Name:

Lost and Found Section

5.9.2 Input:

Items name, description, time of lost/found, picture of the item, map location

5.9.3 Output:

New post will be generate in the lost and found section and visible to everyone

5.9.4 Precondition:

User must be login to the network and account must be activate

5.9.5 Main scenario:

Member A find a notebook in computer lab at engineering building, so he want to post it on the network to find the owner

5.9.6 Step:

Member login to his account
Navigate to lost and found page
Click post found item
Enter the description and other necessary information about the cell phone
Upload the the picture of the cell phone
Enter the location engineering building
Click post

5.9.7 Post-condition:

Other member have the ability to message the original poster
the poster is a friend, they will have access to their phone number,
if not, the available information would be the poster's email address)

5.9.8 Exceptional scenario

Picture size is too large
Location does not identify

5.10 Upload/edit schedule

5.10.1 Name:

Upload schedule

5.10.2 Input:

Course name, time slot,classroom

5.10.3 Output:

Weekly view of the schedule

5.10.4 Pre-condition:

User must be login to the network and account must be activate

User must have their class schedule to upload the information

5.10.5 Main scenario:

Member want upload his schedule with 3 courses

COMP 4770 MWF 9AM TO 10:30 AM,

COMP 4752 MWF 11AM TO 11:50 AM

COMP 4740 MWF 3PM TO 3:50 PM

5.10.6 Step:

Member login to his account

Navigate to profile page

Click schedule link

Choose create new schedule

Enter course name and number(comp 4470)

Pick the day repeated day(monday, wednesday, friday)

Enter time slot of the course (9am to 10:30am)

Click on more course to show more field for next course's information

Repeat for other two course

5.10.7 Alternative scenario:

5.10.8 Post-condition:

Course time slot can not crash other course

Member can view each other schedule

5.11 Upload resume

5.11.1 Name:

upload schedule

5.11.2 Input:

Resume description, resume pdf

5.11.3 Output:

Web view of the resume

5.11.4 Pre-condition:

User must be login to the network and account must be activate

User must have their resume file create

5.11.5 Main scenario :

User want to upload his resume on the social network

5.11.6 Step:

Member login to his account

Navigate to profile page

Click resume link

Choose create new resume

Enter resume description

Click upload resume

Select the pdf file

Click on upload

Repeat for other two course

5.11.7 Post-condition:

Member can view each other resume

5.12 Create Course Poll

5.12.1 Name:

Create course Poll

5.12.2 Input:

course name, option for the course

5.12.3 Output:

Poll view

5.12.4 Pre-condition:

User must be login and activate

User must register to the course

5.12.5 Main scenario: member want to create a course poll about a course

5.12.6 Step:

1. Member login to his account
2. Navigate to profile page
3. Click create course poll
4. Enter course name
5. Enter option for the course
6. Click post

5.12.7 post -condition

Only friends in the same course can respond to the polls posted

Poll owner can modify and delete poll

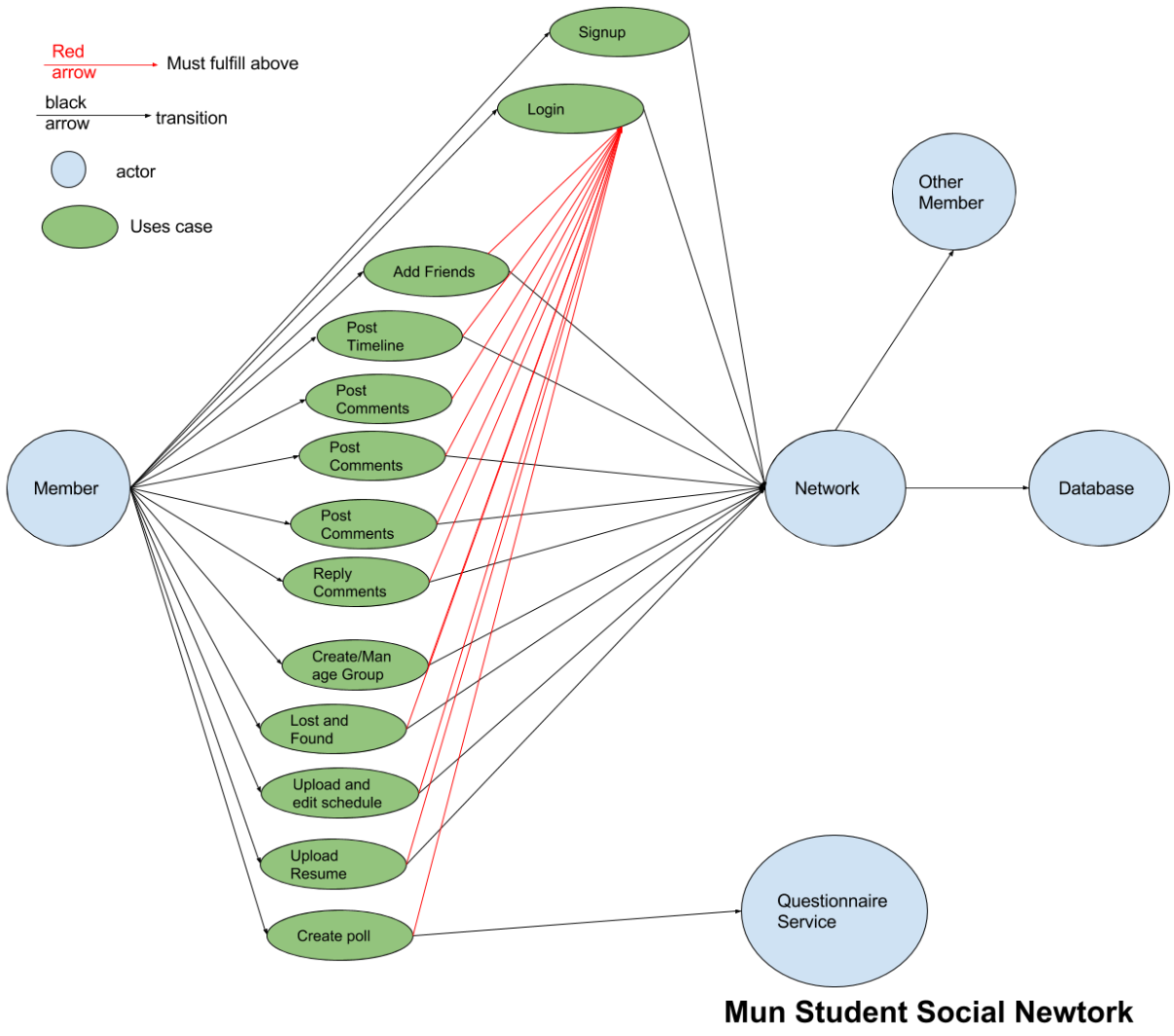
5.12.8 Alternative scenario:

Member wants to modify/edit the poll

5.12.9 Step:

1. Member login to their account
2. Navigate to profile page
3. Click existing poll
4. Click edit
5. Modify the field as necessary
6. Click edit

5.1 3 Use Case Diagram



6 Non functional requirements

Non-functional requirements meet security, usability and other systemic functions not covered in the functional requirements. They govern operations of the system rather than the specific behaviours found in the functional requirements. These are listed below:

6.1 Performance

6.1.1 Response times

The app should take around 10 seconds to load any of the content it has aside from the map, which should take slightly longer, plus it should have a refreshing time of 2 mins when its not being used

6.1.2 Processing times

Most of the processes are simple so they should not take more than 4 seconds regardless of the action

6.2 Capacity and scalability

6.2.1 Throughput

The app should be able to cover around 300 transactions per day when active

6.2.2 Storage

The system should be able to store any amount of text data that the user desires and around 50 pictures per user

6.3 Availability

6.3.1 Hours of operation

The app needs to be available at all times for it to work, it would be down during crashes and upgrades

6.3.2 Location of operation

The app server is located within memorial university server's. Regarding restrictions, a couple node js modules are not available for use

6.4 Maintainability

6.4.1 Architectural standards

The code is going to be divided in different modules, each one for a different purpose. The code itself is going to be organized and will have comments whenever necessary

6.4.2 Coding standards

Organized and readable code that has comments whenever needed to explain functionality

6.5 Recovery

6.5.1 Restore time

The server should be back online instantaneously after crashing

6.5.2 Backup time

It should take around 10 seconds to get the database running and all of the data available

7 Development Tasks

7.1 Webpage Design

Aizaz Iqbal and Keir Strickland-Murphy

7.2 Database

Kristan Hart

7.3 Server

Diego Zuluaga and Meishang Chen