Message Filtering Service

SOFTWARE ENGINEERING seT09102

40214330

# Specification:

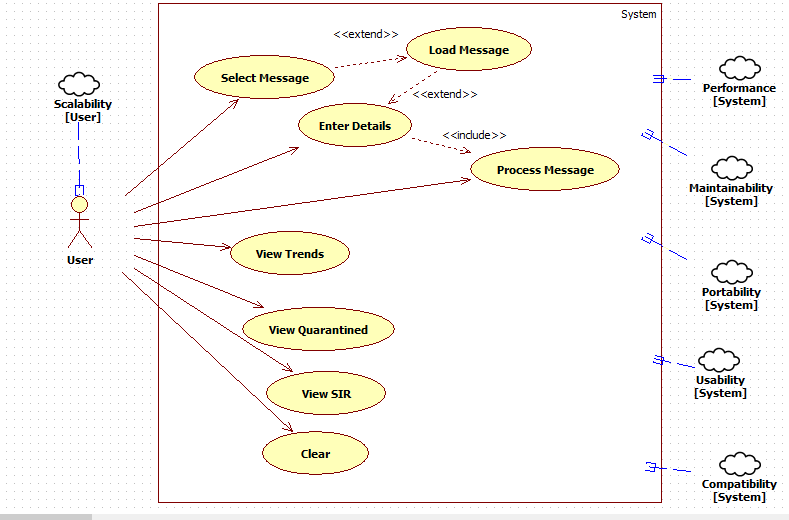
The aim of the coursework was to create a message processing system that would classify automatically the inputs into tweets, SMS or emails. Depending on the type of the message the program will give it an ID, and set limits character limits accordingly for the body of the message. Emails URLs must be quarantined for example, https://www.facebook.com/ will show as <URL Quarantined> and the address will be displayed in a list of quarantined emails (after being processed). Abbreviations in SMS or tweets must be expanded (LOL <Laughing out Loud>). Emails can be of two types standard email messages and significant incident report (SIR). Significant incident reports will be displayed in a list.

Once a correct message is processed it will then output the details as a Json file.

All the hashtags and mentions used in tweet messages are recorded in a trends list to view once the session is finished.

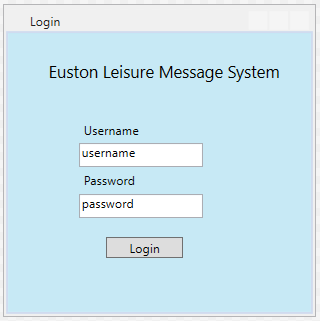
The system also allows to load json or text files that contain the data necessary to fill in the message header and body. It will be then loaded to the system.

# Use Case diagram:



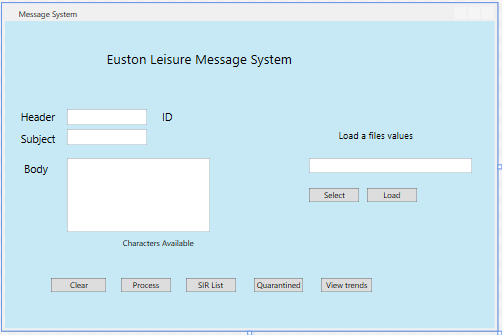
# Design and implementation:

The software was developed in C# using WPF. The program starts with a simple login window (credentials for testing are username = username and password = password):



The message systemform which allows all operations to be done:

It contains the input form, buttons to do each action required and a file loading option.



There is also a message class which holds all the information for each message that will then be used for the processing and importing messages operations.

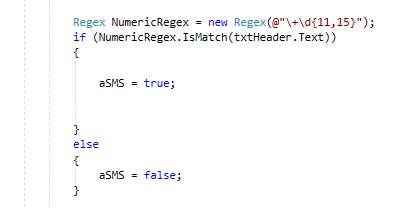
When designing the system, the goal was to keep the message system form simple and easy to use. The software must handle data that is input (either manually or by loading a file) into the textboxes. To ensure that this is done properly the program checks the message type to determine if it’s a tweet, SMS or email. If none is detected, then the message type will be invalid, and the message won’t be processed. This stops the program from serializing each failed message which should never happen as it will just create files and use up resources. When doing the system, regular expressions (regex) were used mainly to validate all the different types of messages and the different requirements for each type of message (such as count of hashtags or mentions, URL quarantined etc)

The txtHeader.TextChanged Event is used to check for a message type each time a character is input into the header textbox (txtHeader). There are specific conditions for each type of message to be able to automatically identify them.

Tweets must start with a “@“ and are of a maximum of 15 characters, international SMS’s numbers start with a “+” and have a minimum of 11 digits and a maximum of 15 digits (range depends on which country code is being used).

Emails just require an email address format such as example.10@gmail.com or any other valid email format.

An example of regex (for SMS type detection), this one checks for a “+” and 11 to 15 digits since the maximum number of digits for an international number is 15 and we weren’t provided a minimum digit requirement.



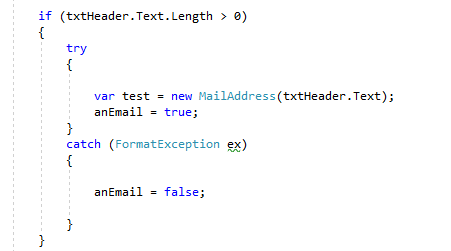
Once the message type is valid it will set a Boolean to true.

There are three Booleans used in the process: anEmail, aSMS and aTweet.

The value of these Booleans is used later to adjust the values of the textboxes such as character limit, width etc. For example, emails are allowed more characters than tweets or SMS’s, so the body textbox will expand to make it more visually appealing.

Email detection is done using the .NET namespace which requires “using System.Net.Mail;”

The following code uses the value of the header textbox and then if it’s a valid email it will set the Boolean as true, else it will remain false.



Once all the appropriate textboxes are filled and the message type has been identified the message will be processed by pressing the process button (using an onClick listener).

When the process button is clicked, the variables from the message class will be given the appropriate values since they will be required for the serialization process.

The software will then check which Boolean is true and process the messages accordingly giving each message type a specific ID name. If all Booleans are false, a message box will show up stating that the type of message is incorrect, and it won’t be processed.

**SMS:**

The only requirement for SMS is that all abbreviations are expanded. We were provided a CSV file containing all the abbreviations and what they should extend to.

For example, LOL should be displayed as LOL <Laughing out Loud>.

To achieve this, I used a dictionary and whenever it detects an abbreviation it will add the necessary text.

**Tweets:**

For tweets the system must record all mentions and hashtags used. To accomplish this task, a regular expression (regex) is used for each type.

This regex matches any string with a ‘#’at the start and adds it to a list called hashtagList. It is all done inside of a foreach loop.

The same is done for the mentions but using a different regex to detect strings starting with ‘@’ which are added to the mentionsList.

The values of these lists are necessary to make the trends lists and display them in the trends form.

Tweets also require abbreviations to be expanded.

**Emails:**

To quarantine addresses the software also uses a regex which will replace the match with <URL Quarantined> and add the replaced match to a list called urlList.

The current text in the body must also be updated after the text being replaced to not display the URL’s anymore.

This is done using a regular expression to match and replace URL’s. Once there is a match it will replace the URL with <URL Quarantined>, add the URL to the list and set the body textbox to the updated text with the URL’s replaced.

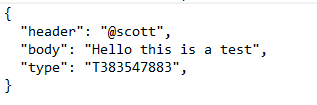
Significant Incident Reports (SIR): These emails have specific conditions, the subject must be SIR mm/dd/yy, the first line of the body must be a SIR Code that follows this example 22-323-22 and the second line of the body must be a specific string for example Staff Attack. Once these conditions are met the values are stored into a SIRemailList to later be used for display.

**Serialization:**

Finally, the last part of the process button is to create the json file with the appropriate data. This process creates a .json file using the type of message ID as a file name.

The data is stored as a json format and it uses the values of the message class which have been set previously. Json uses human-readable text to transmit data objects, which can also be used later to load into the system (deserialization).

This is what a file will look like after being processed by the system (in this case a tweet):



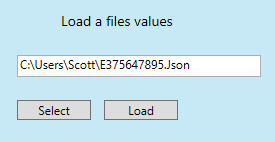
By using this data format, the reverse can be done, retrieving and importing the information into the textboxes automatically. This is called Deserialization.

**Deserialization process:**

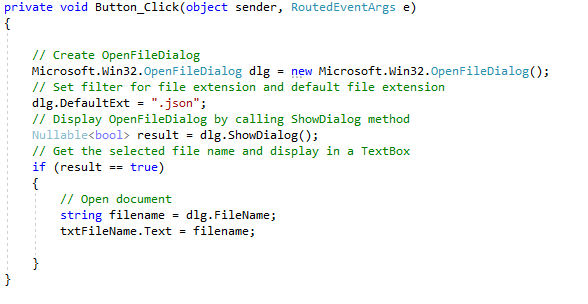
The system allows the user to input files instead of typing in the messages. These files must be either .json or .txt and the data must be formatted as a json data format. If the file doesn’t meet the requirements it will display a textbox with an error.

The system requires the user to use the file selector option or type in the location of the file to input. Once this is done the user must click the load button. Once this is done the json data will be deserialized and the system can process it accordingly.

Example selecting a file



Code required to select a file:



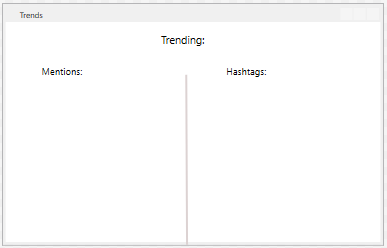
This code will open a file dialog in which the user will be able to browse through files and select one.

The textboxes are set with the data obtained from the file once the load button is clicked. The load button does the deserialize process and gives each textbox the corresponding values.

Load button code, deserializes the object and sets the textboxes and urlList values.



To display the trends list, the view trends must be clicked (background colour was kept white as it makes the text in the display lists easier to read):



**Displaying Tweet mentions and hashtags:**

The design of this form was also kept simple has it just shows two lists and doesn’t have to be interactive.

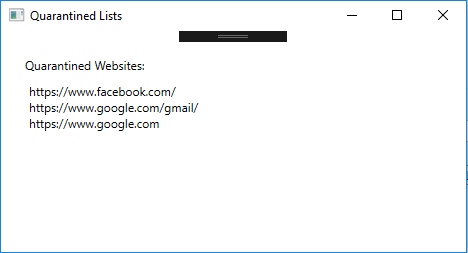
Using the values achieved before that were stored in the lists when processing the tweets, the system counts each time the mention or hashtag is used and then it displays them in its respective text block control in a new form.

This is all done at the end of each session as stated in the specification, and it is displayed once the view trends button is clicked.

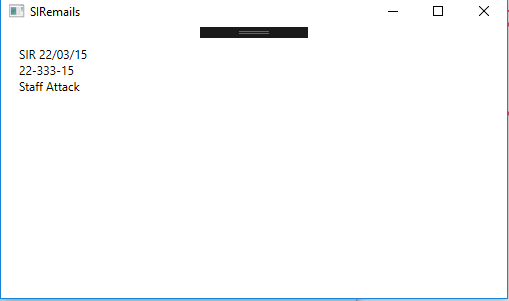


**Quarantined sites and significant incident reports:**

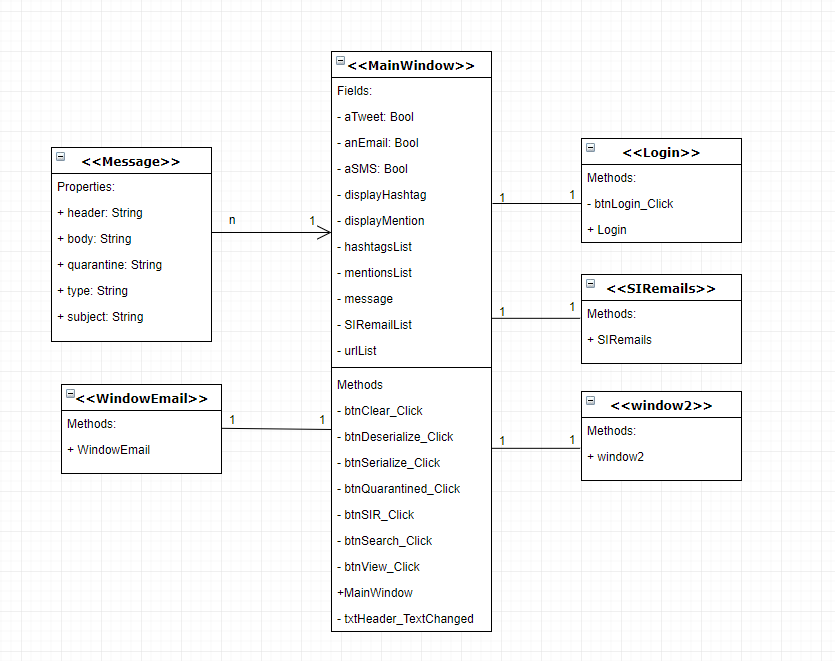
To view quarantined sites, this form will open once the user clicks the Quarantined button. Here all the values of the urlList will be displayed.



This is the SIR Emails List form. All the necessary data for a significant incident report (SIR) will be displayed here. User must click the SIR List button in the main form.



# Class Diagram:



# Version Control:

A version control system is a software package which monitors project files for any changes and allows you to tag the changes at different levels so that they can be visited at any stage, whenever needed.

For this coursework I have used Bitbucket and SourceTree for storing my project on the web as well as having a copy on my own local PC. Bitbucket is a type of Global Information Tracker (GIT).

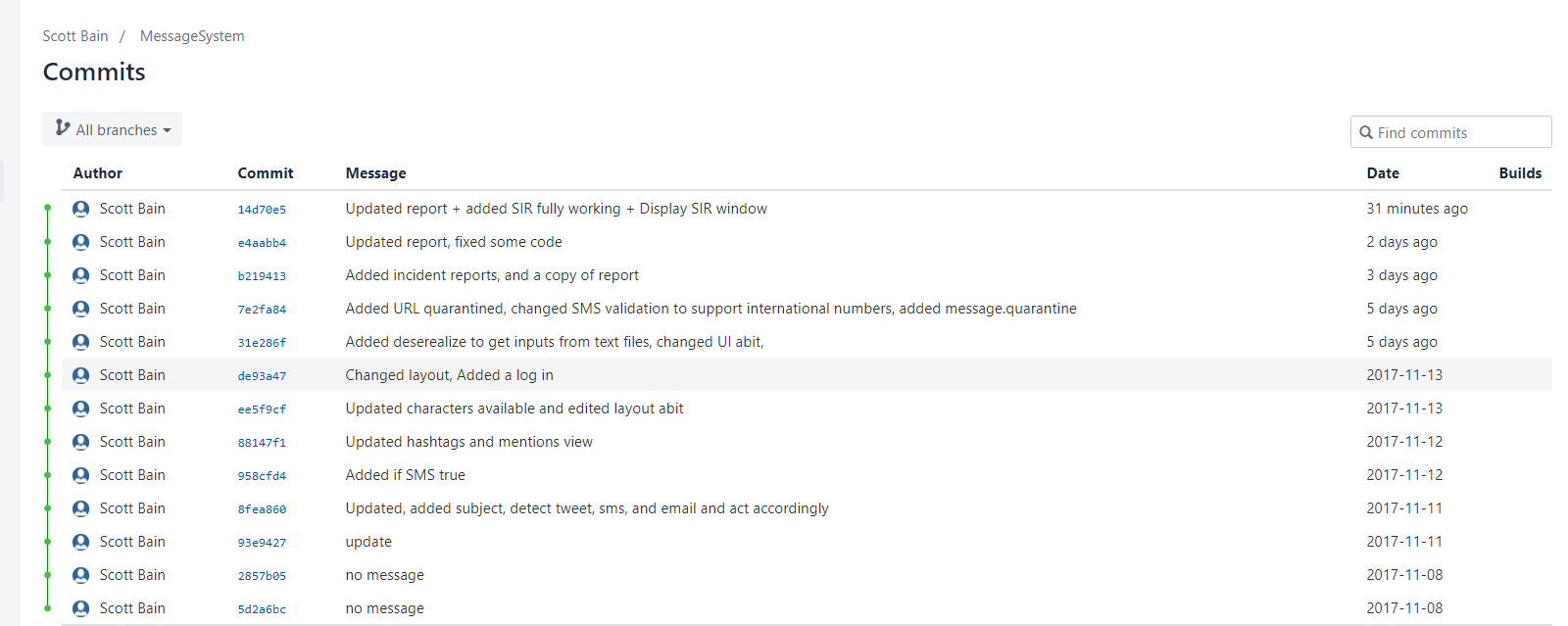
The main advantage of using any type of git repository such as Bitbucket, is that it allows to easily roll back to your previous version and you will always have a backup available online.

It is also good for working in teams and sharing a repository which wasn’t the case as this is an individual coursework. Files can also be removed from the repository.

Each person of the team is assigned with doing certain tasks and they can all share the repository and keep up to date with all changes made to the software.

Every team member has access to the source code and they can be given certain roles such as read-only, administrator or be allowed to write.

Here’s a screenshot of my first commits to Bitbucket for this coursework.



Each time the project has gone under changes and it is saved, a new commit is made to also save it in bitbucket. A commit records changes to the repository.

The message is used to explain what the changes were to easily keep track of what was each commit. Pressing the commits allows you to rollback or see the previous stages of the files. It also provides the date for each commit.

# Testing Plan:

When developing the software many tests had to be done. To keep track of how the system reacts to certain inputs I recorded many different tests to ensure all functionality is working as intended.

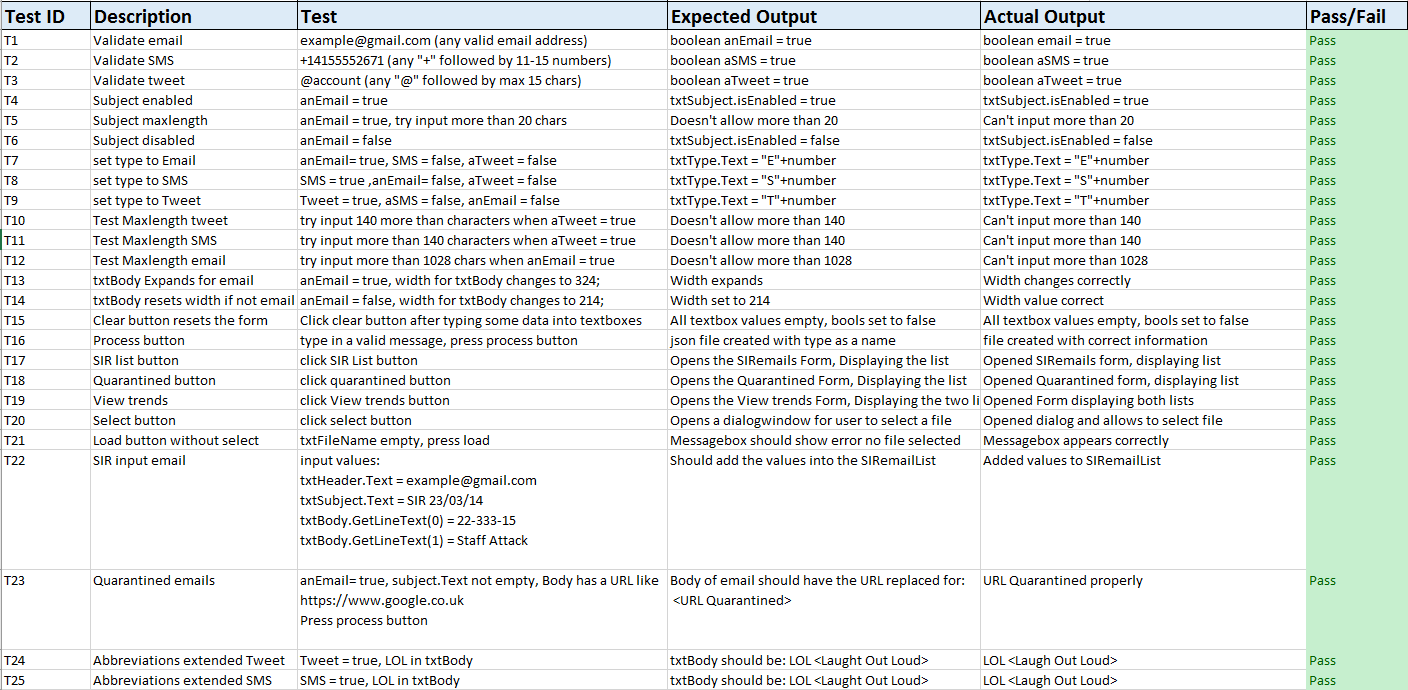
Having a testing plan also helped identify if the system needed any types of limitations for example limiting the regex for the SMS to detect only 11-15 digits.

Having records of these tests saves time whenever checking functionality of the system as you can keep track of the parts of the code that have already been proven to work as expected.

Also, having a list with tests that failed can ensure that any found bugs are addressed and fixed at some point in the development. Software should be released once it has passed all the tests.

Since the requirements were mainly validating inputs from the textbox and changing the settings for each control (maximum character length for example), or giving specific ID’s, the testing was helpful to minimize the time checking the functionality and typing information in the textboxes.

This is my testing records for the Message System Application



# Evolution Strategy:

Software change is inevitable, new requirements often emerge when the software is being used or for other reasons such as environment changes, errors that must be repaired, new systems etc.

Software evolution is important because organisations have huge investments in their systems and companies rather updating and evolving the current software rather than developing new software.

Maintenance is also important and there are different types:

-Corrective: to repair software faults.

-Adaptive: to adapt software to a different operating environment.

-Perfective: to add or modify the system’s functionality.

Systems must be maintained to ensure that everything works as intended. Maintenance doesn’t have to be done by the developer and is often expensive depending on the type of software.

For the messaging system some new features might need to be added.

For example, twitter has changed their 140-maximum character limit for tweets to 280, meaning that the system is currently outdated and needs to be changed.

New features for the different types of messages might need to be implemented, or more options such as add a filter for types of incidents for the Incident Report List. This would allow users to easily add any new types of incidents to the system that were not defined in the current software requirements.

With the rise of new social medias or messaging systems, the company might decide that they want to upgrade their system to be able to stay relevant with the new trends.

Since the current system isn’t very complex it allows to easily implement new features or fix any possible issues that may occur. For bigger projects making changes might affect the overall functionality and making a small change could cause bigger problems.