University of Glasgow | School of Computing Science

Honours Individual Project Dissertation

Event Tracker: A Social Media Analytics Platform for Use During Disasters

Charles Thomas

March 23, 2019

Abstract

The tools available for emergency management organisations to successfully monitor and manage events during crisis situations are lacking. Current systems have been ineffective due to the lack of training and guidelines for new staff members, along with questions over the data quality and trustworthiness. Event Tracker is a new solution providing a unified view of a crisis event, designed to support emergency response agencies, volunteers, and the victims during a disaster. In particular, Event Tracker provides several novel functionalities, including the generation of actionable information feeds, criticality alerting, and response agency and volunteer management. Following an extensive user evaluation, it was found that with these functionalities, Event Tracker could prove highly effective in supporting these organisations and aiding the response effort during a deployment. Furthermore, a demonstration paper regarding the new solution has been submitted to SIGIR 2019, a CORE A* conference aimed at the research and development of information retrieval.

Acknowledgements

I would like to thank my supervisors Professor Iadh Ounis and Dr Richard McCreadie for their invaluable support and guidance throughout the year. I would also like to thank the user evaluation participants for their time and input, which was extremely valuable.

Education Use Consent

I hereby grant my permission for this project to be stored, distributed and shown to other University of Glasgow students and staff for educational purposes. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Signature: Charles Thomas Date: 23 March 2019

Contents

1 Introduction

This chapter will introduce the Event Tracker project, examining the motivation and aim behind the development of the system. Further chapters in this paper will go on to detail specific areas of the project, including its background, design, implementation and evaluation.

1.1 Motivation

A 68% rise in the use of social networking sites since 2005 (Perrin 2015) has introduced an abundance of real-time information online. This has enabled new ways for the public to contact response agencies (Castillo 2016) and grants those in the emergency management sector a new means of accessing potentially life-saving information. With a three-fold increase in natural disasters over the past 35 years (Hoeppe 2015), it is extremely important that emergency response agencies have the tools available to ensure that they can monitor social media streams in real-time and aid the public in a quick and effective manner.

Data posted on social media platforms regarding a disaster or world event could potentially provide a wide range of valuable information to emergency response services. For example, responding to cries for help, accessing first-party observations or simply gaining an insight into public opinions could provide an added-value to these agencies. However, with the wealth of information now available, it is critical that the data can be easily visualised and navigated, ensuring that emergency services can quickly find and act upon key information.

There also lies potential in the connection of volunteers with formal emergency response officers during crisis situations. Systems designed to assist emergencies services during crises typically face a number of challenges, including a lack of coordination and communication between the officers and other formal entities as well as an unwillingness to engage and form relationships with digital volunteer groups (Twigg and Mosel 2017). A successful connection with both physical and digital volunteers could indeed aid response efforts immensely, with a wealth of potential skills and resources becoming available (Whittaker et al. 2015).

A range of systems have been developed in the past with the goal to support emergency response efforts during disasters (Castillo 2016). These systems, however, have had little impact on the sector. It has been found that reasons behind this include the insufficient training of personnel, which require time to make adequate use of the platforms, the lack of guidelines for their use and apprehension over social media trustworthiness (Hiltz et al. 2014). There is potential in the development of a new system which could combat the common downfalls of past attempts and successfully aid emergency response efforts.

1.2 Aim

This paper proposes Event Tracker, a new system which aims to support the monitoring and management of events during both past and ongoing crisis situations. The system intends to support three primary tiers of users:

1. Emergency response officers and their teams
2. Physical and digital volunteers
3. Victims during the crisis

Event Tracker aims to provide emergency response agencies with the tools to navigate large volumes of social media data with ease, supported by several key functionalities. These include the generation of actionable information feeds, criticality estimation and on-request information access. Volunteers should be able to provide information directly to these response agencies, either using their first-person knowledge of the situation or aiding the data navigation and highlighting specific information and relevant details. Victims during a disaster should be able to straightforwardly access advice and guidelines from emergency services, ensuring that they are equipped with up-to-date information that could lead to their safety.

The proposed system should be built using a flexible and modular architecture, designed to provide automated, low-latency ingestion and augmentation of report streams. During a disaster event, a high rate of relevant content may be required to be processed - potentially up to 4000 posts per minute (Ounis et al. 2016). Hence, the system is required to handle such volumes of data efficiently without compromising user experience. Event Tracker will make use of existing services for both the ingestion and augmentation of data, and so modularity is required in the case where services may be added or removed in the future. The frontend of Event Tracker must also be designed in such a way to support different modules, ensuring new functionality can easily be added without requiring a full redesign. Services available for use in the system are examined in further detail in Section 2.2 of this paper.

Event Tracker should be a modern, responsive web application, enabling a high level of user interaction. The system should be accessible on any device, such as a desktop or mobile, with no compromise in features on either device. It is key that all user types specified can use the application on a mobile device as it may not be possible to access a desktop during a crisis situation.

1.3 Summary

This chapter outlined both the motivation and aim behind the development of Event Tracker to make clear the overall objective of the system, along with how it aims to add benefit to the emergency management sector. The remainder of this paper will discuss how the motivation for Event Tracker was developed into a working system, and how the aims were met. The paper is structured as follows:

- **Chapter 2** researches related products to Event Tracker, identifying their strengths and weaknesses. This chapter also investigates the backend services available for use throughout the project.
- **Chapter 3** outlines the requirements analysis process, from creating user stories and scenarios to developing a complete set of functional and non-functional requirements.
- **Chapter 4** discusses the design process including the system architecture, tools and technologies to make use of and constructing the user interface, relating each design decision back to the project requirements.
- **Chapter 5** reviews the implementation of Event Tracker and highlights the software engineering processes involved throughout the project.
- **Chapter 6** provides an evaluation of the system, including unit testing, performance evaluation, user evaluation and requirement validation.
- **Chapter 7** briefly discusses the demonstration paper about Event Tracker, which was submitted to SIGIR 2019.
- **Chapter 8** summarises this paper, examines any future work which could be carried out and provides an overall reflection of the project.

2 Background

This chapter will examine the background of the project, first by looking at several related products that have similar goals or functionality as that of Event Tracker. The chapter will then investigate the pre-existing services which form the foundations the system is built upon.

2.1 Related Products

A range of systems created for both academic and commercial purposes have been introduced with a similar goal as that of Event Tracker in supporting emergency response efforts during crisis situations (Castillo 2016). This section provides a review of several of these products, identifying any strengths or weaknesses and discussing how each has influenced the design and development of the new platform.

2.1.1 Emergency Analysis Identification and Management System

The Emergency Analysis Identification and Management System (EAIMS), developed at the University of Glasgow, was a prototype crisis management system that aimed to exploit social media data to provide real-time detection of emergency events, along with search, sentiment analysis, discussion-thread extraction and summarisation functionalities (McCreadie et al. 2016).

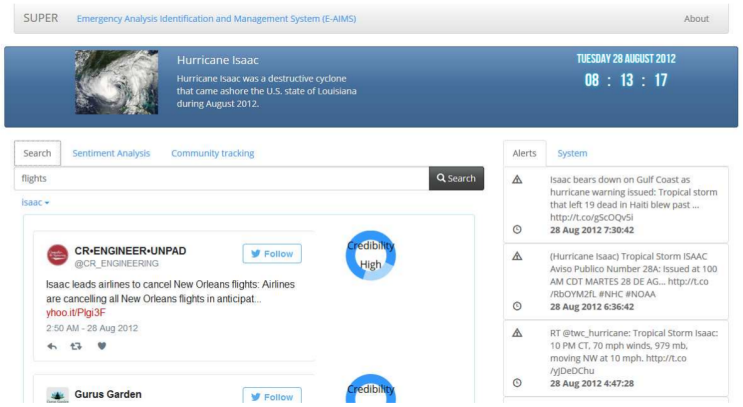


Figure 2.1: *Section of the EAIMS user interface, displaying the results following a user search.*

EAIMS presents a lot of functionalities through the use of its backend services, each of which adds a new element to the system. One important service is *event detection*, which automatically finds the first instance of a post that represents a new event. These can then be filtered to extract potential events of interest to the user, allowing the detection of world events and disasters quickly. EAIMS is also integrated with several tactical services, for instance, *integrated search*. This is highly

optimised, allowing users to quickly search and retrieve social media content relating to specific world events.

The main drawback to EAIMS is that it was primarily designed for use by high-level, head-office, emergency response staff. It is apparent that whilst the system could prove very beneficial for these users, a lot of the information provided could also be useful to other users. In contrast, Event Tracker aims to close this gap by focusing on a greater user base, including physical and digital volunteers, as well as the victims during a disaster.

2.1.2 Twitter Moments

Twitter Moments is a relatively new feature of the Twitter platform, added in 2015 (Alton 2018). A 'Moment' is a user-curated collection of Tweets, which allows users to comment on stories, promote news or create round-ups of different world events. Reen (2018) recommends the use of Twitter Moments during emergencies and natural disasters, as users can easily discover information about the ongoing emergency by following the Moment and receiving crucial updates as they are added.

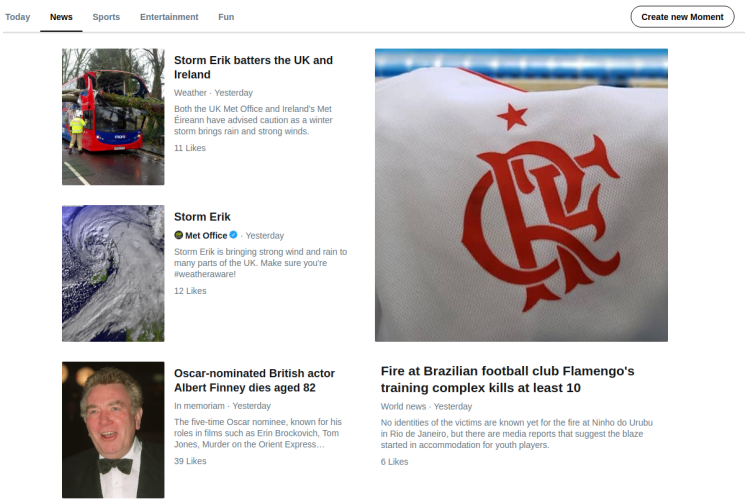


Figure 2.2: *Twitter Moment's user interface, where user-curated collections of Tweets can be viewed on many topics.*

Although Twitter Moments has great potential in aiding emergency response agencies and those caught up in a disaster, a key downfall is that Moments are user-curated and popularity-driven, and so critical content such as individual calls for help will be missed. As the feature can also be used to track other news such as sports or entertainment, as shown in the user interface in Figure 2.2, Twitter Moments is not an ideal system for tracking disasters. Instead, Event Tracker aims to be strictly focused on crisis situations to aid response efforts and relies on recall-focused crawling in conjunction with automatic content categorisation to find actionable information quickly and accurately. This should ensure that all relevant content is processed, and user curation is not relied upon.

2.1.3 Artificial Intelligence for Disaster Response

Artificial Intelligence for Disaster Response (AIDR) is a platform developed at the Qatar Computing Research Institute that collects and classifies crisis-related Tweets (Imran et al. 2014). AIDR

makes use of crowd-sourcing to label Tweets, which can then be used to train an automatic classifier. This ensures the system performs well for each new event registered, as automatic classification using only pre-trained models can lead to low accuracy.

Currently, Event Tracker relies on pre-trained classification models rather than crowd-sourcing. In the future, if it is found that this method is leading to low accuracy classification, due to the flexible underlying microservice architecture, the integration of services such as AIDR would be possible. Additional services could be easily added or removed, allowing services to be compared effectively through the Event Tracker system.

2.1.4 Twitcident

Twitcident is another web-based system for filtering and analysing social media data on real-world incidents, such as disasters (Abel et al. 2012). Like EAIMS, Twitcident provides incident detection and search technologies, enabling emergency services to filter data to aid their response efforts. However, Boersma et al. (2016) state that Twitcident did not provide much value to response agencies in practice, as it could not provide early warnings regarding critical information. For this reason, Event Tracker integrates criticality estimation tools to provide real-time reporting of crucial information.

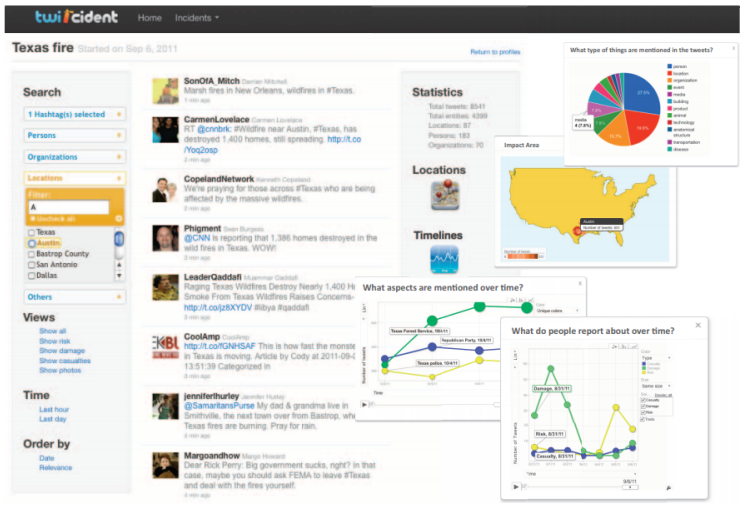


Figure 2.3: *Twitcident platform, being used to analyse social media content relating to the phrase 'Texas fire'.*

2.2 Project Foundations

As discussed, Event Tracker's flexible architecture will allow for the integration with several pre-existing services, each designed to increase the range of capabilities of the system with the greater goal of improving emergency management. These services provide functionality such as report classification and on-request information access. This section investigates the services which were integrated with Event Tracker over its development.

2.2.1 TREC Incident Streams

One key reason which has led to previous systems making little impact in the emergency management sector is data quality and the inability to get the key information to the right people (Hiltz et al. 2014). To combat this, Event Tracker will make use of a state-of-the-art automatic content-based classification service which will enable social media posts to be grouped into specific event-related categories to aid data navigation. As well as content categorisation, the classification service can also be harnessed by Event Tracker to provide real-time criticality estimation, designed to predict high priority reports enabling emergency response officers to quickly identify potentially important information.

The classification service targets a crisis ontology which is being maintained by TREC Incident Streams (TREC-IS). TREC-IS is a new initiative at the Text Retrieval Conference designed to solve data quality issues by developing test collections and evaluation methodologies for filtering approaches which identify and categorise aid-requests posted on social media networks over the course of a crisis event (McCreadie et al. 2019). The TREC-IS track introduces a bi-yearly evaluation challenge aimed to increase the technology readiness level of current social media crisis monitoring solutions, and hence in the future, the classification for Event Tracker should only get better.

2.2.2 Integrated Search

With the high volumes of data collected over an event, it is imperative that response agencies can easily navigate the wealth of information, therefore the ability to search over the collection is a key feature which can add a lot of benefit to Event Tracker. The proposed system will make use of the same integrated search service as EAIMS, originally developed as part of a larger system, SUPER (SUPER 2017). SUPER is a European Commission project for emergency management applications to collect and process social network data to produce useful information and provide services to end-users such as emergency response agencies (Moralis 2016).

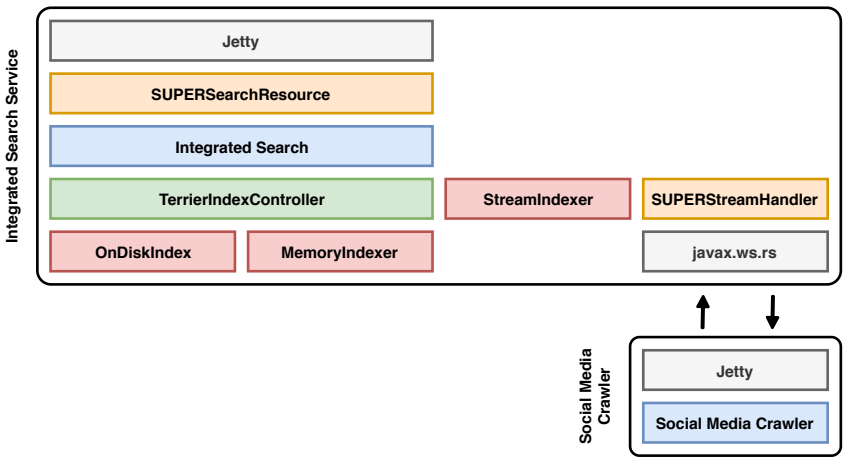


Figure 2.4: *Integrated search service architecture, displaying the connection with the SUPER social media crawler and underlying Terrier search engine platform. Figure adapted from Ounis et al. (2016).*

The architecture for the service is pictured in Figure 2.4, where it can be seen that the service is built on top of the Terrier search engine platform (Ounis et al. 2006). Terrier handles the real-time indexing and information retrieval functionalities present in the service (McCreadie

et al. 2016). The indices support highly efficient data retrieval, making high search speeds over large volumes of data possible - ideal for Event Tracker.

2.2.3 Social Media Crawler Streaming Client

In order to both develop and effectively evaluate the proposed system, it would not be viable to use an active real-time social media crawling solution to gather reports relating to ongoing disasters. This could lead to irregular data streams making it difficult to test out the functionality provided. For this reason, a past event report streaming simulator was introduced to the project.

Event Tracker will make use of the Social Media Crawler Streaming Client (SMCSC) outlined in Kardara and Violos (2015). Like the integrated search service, the SMCSC was developed as part of the SUPER middleware platform. The SMCSC was developed to simulate streaming of data from a social media crawler, previously developed for the SUPER platform, as the crawler can only be accessed through a REST API, which is not suitable for real-time data streams.

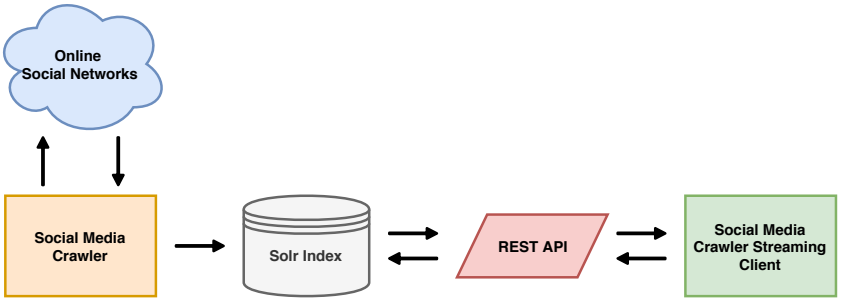


Figure 2.5: *Social Media Crawler Streaming Client data flow, highlighting the architecture between the online social networks and the SMCSC. Figure adapted from Kardara and Violos (2015).*

Figure 2.5 illustrates the data flow within the SMCSC. There are three main processes: collection, indexing and retrieval. The social media crawler component makes use of public social network APIs such as Twitter, and then sends the crawled data to be stored in an Apache Solr index (Apache 2019). The REST API can then be used to retrieve data from the Solr instance. The SMCSC makes periodic calls to the REST API and adds returned JSON data to a stream, implemented as a blocking queue.

Event Tracker can harness the SMCSC by sending a request query with a unique event identifier string, and receiving a pseudo-real-time stream of reports, each representing a social media post regarding the relevant event. Event Tracker can then process and display these reports in different ways, each designed to assist in the navigation of the high volume of data, to aid emergency response agencies and volunteers alike.

2.3 Summary

This chapter reviewed multiple existing products, both academic and commercial, and how each of these influenced Event Tracker in some way. Work completed prior to the development of this project was also examined, looking into components such as TREC-IS and the integrated search service, and how these technologies were to be integrated within the new system.

3 Requirements Analysis

This chapter will discuss the project's functional and non-functional requirements and the methods in which these were elicited. Requirements were determined during the beginning phase of the project, however, as the project progressed and more services were implemented, they were later revisited and updated accordingly.

3.1 Requirement Elicitation

In order to devise the requirements of the Event Tracker system, multiple user scenarios were initially created to aid in describing the context of how different users would interact with the application. These scenarios were then converted into user stories, allowing for a high-level description of the requirements.

3.1.1 User Scenarios

Multiple user scenarios were developed, describing the ways in which different types of user may interact with the system, including system administrators, emergency response officers, volunteers and victims. The following scenarios all centre around an earthquake which will be tracked on Event Tracker.

- **Lucy** is an admin on the Event Tracker system. One night whilst watching her dashboard she notices a large earthquake has just struck in Barcelona, with more and more Tweets reporting the event every minute. Lucy quickly sets up multiple different groups on the system that she knows will be required, such as the fire service, ambulance service and the Red Cross. She looks up who would oversee these and assigns a leader to each group. An open volunteering group has already been created for the event, which is starting to gather members. Lucy can't help much more from London with the earthquake, however, she is confident that the different groups assigned to the event will be doing as much as possible to help. She can keep an eye on the event to accept any new incoming group requests for different response agencies she may have missed.
- **Mateo** is part of the Barcelona Fire Service. He has retired from fighting fire, and tends to stick in the office, aiding the firefighters by giving them updates with crucial information as it comes in. Whenever a nearby disaster happens, he is assigned as the leader of the fire service group on Event Tracker. He first sets up his agency by adding other users to his organisation and then uses the system to view real-time reports regarding what the public are posting about the disaster on social media. An earthquake has struck so he is on constant watch of the system, watching new data stream onto the feeds on his dashboard. He can use the criticality altering system to become immediately aware of potentially crucial information, which he can then direct to whoever will be able to deal with it best. The members of his group will be aiding him by also highlighting Tweets which may be of importance. Mateo notices a tweet, highlighted by a volunteer on Event

Tracker, from a British tourist whose hotel has collapsed, and she does not know where to go. He can then get in contact with the Red Cross group, who will be able to help the tourist. Once the aftermath of the earthquake has settled, Mateo generates a summary of the event, containing details such as the timeline of important incoming tweets, and how many tweets were made associated with emerging fires throughout the city.

- **Camila** is a nurse from Barcelona who has been a Red Cross volunteer for two years. When an earthquake hit in her home city, she knew she would do anything possible to help. The incident commander for the Red Cross added her to the organisation's group on Event Tracker, so she can get updates on what is going on and who needs help. She notices a post that someone from the fire service had created regarding a British tourist whose hotel had been destroyed in the disaster and did not know where to go. The Red Cross had opened up temporary shelters for stranded civilians, and so Camilia can use the crisis mapping functionality of the system to find out exactly where the Tweet was posted. She then can create a publicly available notice on the Event Tracker system to say where the closest shelters were, and that anyone stranded should head there for safety.
- **Jane** is an accountant from Glasgow, who was on a holiday visiting Barcelona. An alarm in her hotel went off to say an earthquake had struck, and so she left to get to safety. She was unsure what to do and sent a desperate tweet to her followers asking for help. She then went on Event Tracker, where she saw a notice from someone in the Red Cross saying that a shelter for the stranded had been set up close by, and so Jane could head there for safety.
- **Samuel** is a shop owner from Barcelona, who was devastated when an earthquake had destroyed so much of his city. He wanted to help but didn't know how so went on the Event Tracker system where he saw he could join an open volunteering group. He runs several searches on his dashboard to try and find important information and finds a Tweet from a tourist who is stranded. He uses the system to highlight this Tweet to the emergency response teams who are better suited in helping the tourist. He then customises the different Tweet categories which are displayed on his dashboard to find requests for donations, goods and services. He finds a Tweet requesting for any supplies such as food or water to be brought to temporary shelters set up around the city. Samuel knew he was in a position to help, so emptied his shop shelves into containers and brought the supplies to the shelter.

3.1.2 User Stories

Following the conception of the user scenarios described in Section 3.1.1, different levels of user were decided upon and user stories were created for each. The results of this are listed below.

Event Tracker User

- *As a* user on the system, *I want to* register and login to an account, *so that* I can personalise my experience.
- *As a* user on the system, *I want to* view current events on my dashboard, *so that* I can quickly see all details of an event.
- *As a* user on the system, *I want to* filter which events are shown to me, *so that* I do not have to see any irrelevant data.

Event Tracker Admin

- *As an* admin, *I want to* create different groups associated with each event, *so that* there is a group for each organisation involved.
- *As an* admin, *I want to* assign group leaders to different groups, *so that* there is someone in charge who can invite members to the group.

Group Leader

- *As a* group leader, *I want to* assign multiple users to become a member of my group, *so that* they can represent my organisation.
- *As a* group leader, *I want to* view a subset of details of an event, *so that* I only see the relevant information I need.
- *As a* group leader, *I want to* be alerted with reports of potentially important information, *so that* I can respond quickly.
- *As a* group leader, *I want to* generate event summaries/reports, *so that* I can pass this information on to anyone applicable.

Group Member

- *As a* group member, *I want to* request to join the group I am associated with, *so that* I can represent my organisation and receive important information.
- *As a* group member, *I want to* add notes to the event I am dealing with, *so that* I can add important information to others.
- *As a* group member, *I want to* communicate with others on the system, *so that* I can send and receive important updates.
- *As a* group member, *I want to* access crisis mapping tools, *so that* I can see the origin locations of different data reports.
- *As a* group member, *I want to* be able to filter and customise different information feeds, *so that* it is easy to navigate through large collections of data.
- *As a* group member, *I want to* search over the collected data, *so that* I can quickly find information relevant to my needs.

Volunteer

- *As a* volunteer, *I want to* join an open volunteering group, *so that* I can help in any way possible with the information I have.
- *As a* volunteer, *I want to* connect and communicate with emergency response agencies, *so that* I can aid their response efforts.

Victim at an Event

- *As a* victim at an event, *I want to* quickly see information about said event, *so that* I can receive advice about how to get to safety.

3.2 Functional Requirements

Once the requirement elicitation process was complete, a set of functional requirements could be established. These requirements would then be used as a reference throughout the project, to ensure the system met all of its initial criteria. The MoSCoW method (Consortium 2014) was used, which is a method of prioritisation in which requirements are labelled as one of the following:

- *Must Have*, which describes requirements which are critical in the success of the project, where failing to meet these will result in a solution which is not viable.
- *Should Have*, which describes requirements that are important, but the success of the project does not depend on their completion.
- *Could Have*, which describes requirements which are desirable, but it would be acceptable if they were not present in the final system.

- *Won't Have This Time*, which describes requirements that will not be delivered, but are useful to state in order to make clear the scope of the project.

Like the user stories in Section 3.1.2, the functional requirements have been separated into user types so throughout development a single user type could be focused on at one time. As group leaders should have access to similar functionalities as group members, these two user types have been joined into a single *Emergency Response Groups* type. Requirements are listed below, along with their importance, represented by **MH**, **SH**, **CH** or **WH**, which corresponds to the *Must Have*, *Should Have*, *Could Have* and *Won't Have This Time* labels respectively. Requirements marked with ***** were added throughout the project as different iterations introduced possible new functionalities.

Event Tracker User

1 – MH Users must have the ability to create an account and log into the system.

2 – MH Users must have the ability to see current events on a dashboard.

3 – MH Users must be able to switch between events to view only relevant data they are interested in.

Event Tracker Admin

4 – MH Admins must have an admin-focused dashboard containing an event feed.

5 – MH Admins must have the ability to create different groups representing different emergency response agencies.

6 – MH Admins must have the ability to allocate users to be group leaders to the groups they are creating.

7 – SH Admins should have the ability to accept incoming group requests for new groups related to an event.

Emergency Response Groups

8 – MH Group leaders must have the ability to see that they are in charge of a group of users representing their organisation.

9 – MH Group leaders must be able to assign other users to be part of their group.

10 – MH Groups must have a dashboard containing multiple feeds on data associated with the event they are assigned to.

11 – MH Groups must be able to add notes to current events and communicate with other users on the system.

12 – SH Group leaders should have the ability to request that their group is added to an event to represent their organisation.

13 – SH Group leaders should have the ability to broadcast information to the public view of the system.

14 – SH * Groups should have the ability to filter incoming data by customised categories depending on their needs.

15 – SH * Groups should be able to view predicted criticalities of incoming data reports.

16 – SH * Groups should be able to search over the data collected for an event, and filter results by date.

17 – SH * Groups should be alerted when new reports are predicted as critical, so they can act upon the information quickly.

18 – CH Groups could be given crisis mapping tools to locate where different reports have originated from.

19 – CH Group leaders and members could receive notifications about changes to their group or new posts.

20 – CH Group leaders could have the option to generate an event summary of their associated event.

21 – CH Groups could have the ability to view a timeline of event highlights, during or after an event.

22 – WH It was decided that Event Tracker will not require a fully functioning real-time messaging system, as this would be a large addition to the scope and does not provide enough benefit in completing the aims described in Section 1.2.

Volunteer

23 – MH Volunteers must have the ability to easily join an open volunteering group, where there is no set group leader thus there is no requirement to be accepted.

24 – MH Volunteers must have the ability to communicate with emergency response agencies to aid response efforts.

25 – SH * Volunteers should have the ability to filter data by categories which are relevant to them, such as 'Donations' or 'Volunteer Requests'.

Victim at an Event

26 – MH A victim at an event must be able to view details about the event without requiring an account on the system to deliver information as quickly as possible.

27 – MH A victim at an event must have access to up-to-date guidance with instructions on how to proceed in their situation.

3.3 Non-Functional Requirements

Along with functional requirements, a set of non-functional requirements were established before any further progress of the project could take place. Whilst functional requirements describe what the system should do, non-functional requirements are equally important as they describe the properties that the system must have (Robertson and Robertson 2012). All established non-functional requirements were Must Have requirements, as they are all essential in a successful product.

28 – MH The system must be modular, allowing for different services, feeds and visualisations to be added or removed in the future.

29 – MH The application must be responsive and should function correctly on both desktop and mobile devices without any problems or restriction of features.

30 – MH The system must be portable and work on all modern browsers.

31 – MH The system must be intuitive and easy to use by untrained users.

32 – MH The system must perform well, with quick load and response times, allowing for better a user experience.

33 – MH The system must be made possible to integrate with other EPIC-Sys components, as per customer request.

3.4 Summary

This chapter detailed the different scenarios in which the Event Tracker system could be used, along with the different user types the system will represent and their associated user stories. It described how from these a set of functional and non-functional requirements were created, and how the MoSCoW prioritisation technique was implemented to structure these requirements and identify their importance.

4 Design

This chapter will discuss the design of the product, starting with the architecture the system was based upon. The user interface design for the system will also be discussed, where the different functionalities will be described and each be linked to the requirements they satisfy. Finally, the chapter will then go on to examine the tools and technologies that were considered for the project, and of which were used throughout implementation.

4.1 System Architecture

The Event Tracker system can be broken down into two main parts. These are the web application and the backend services. As explained in Section 1.2, the project aim was to build Event Tracker over existing backend services, with no new services requiring implementation. Thus, the focus of the project, and hence the architecture, was the web application and its interaction with the services.

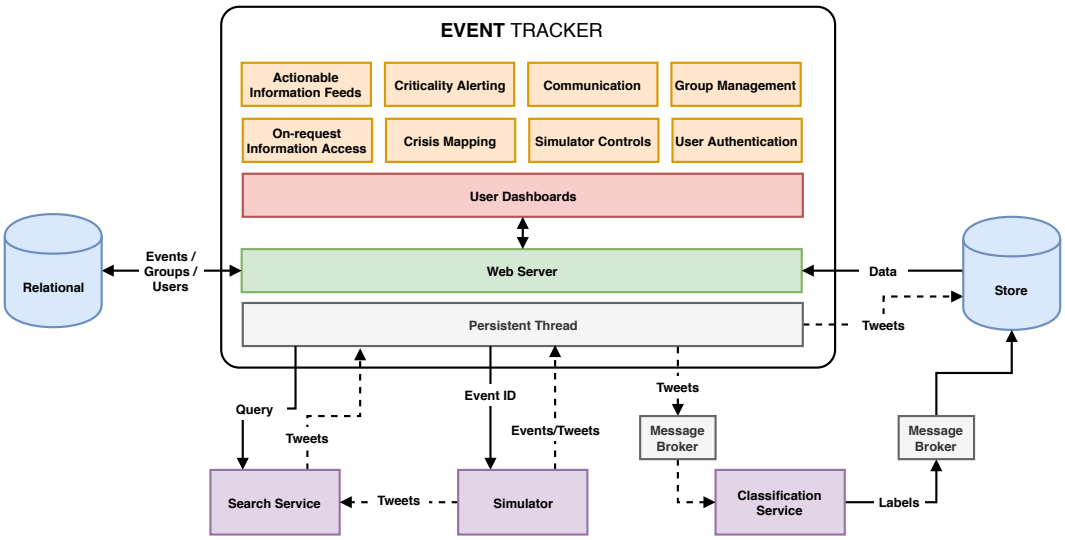


Figure 4.1: *Full system diagram, describing the connections between each component of the system.*

The system was designed following a three-tier architecture pattern (JReport 2018), consisting of a presentation layer, an application layer, and a data layer. The reasoning behind this key decision was that for the system to successfully meet requirement **#28**, it had to be modular and flexible. This architecture granted the ability to focus on each layer separately, where no change in a single layer could introduce a defect in another. The **presentation layer** simply contains the user interface of the web application and was required in order to visualise data to be shown to the user and allow for user interaction with the system, vital for many of the project requirements. This layer interacts with the web server in the application layer through RESTful

queries. The **application layer** contains the logic for the system and thus the fundamental core of Event Tracker. This layer includes the integration of the pre-existing services, which is further examined in Section 5.2. Lastly, the **data layer** comprises of all the data storage for the system. The setup for this layer had to be considered carefully, as a potentially large volume of data will be stored, whilst still ensuring data could be accessed with low latency.

In order to fully understand the system architecture and how each component should interact with each other, Figure 4.1 was created. This describes the connections between the elements, along with the type of data being transferred. The web server acts as the core of Event Tracker, bridging the gap between the microservices and the user interface. The user interface and associated modules are discussed in the following section of this paper. It was decided that to achieve the best performance whilst still allowing for high-volume, low latency data transfer, the data layer would be separated into two independent database instances. Indeed, a relational database is required to handle all internal Event Tracker data, such as events, users and groups. Furthermore, a second database, hereinafter referred to as the "store", will store any Tweets or other relevant service data, such as classification labels. A persistent thread is set up to run on the server which handles the interaction between both the simulator and the services with the web application. As discussed in Section 5.2.1, this was added during the implementation process to ensure communications between Event Tracker and the services would always run to completion, even if the originating browser session is closed. A message broker was introduced to the system to handle interaction with the classification service - the set-up for the broker is further examined in 5.2.2.

4.2 User Interface

A high number of the functional requirements involve some required user interaction with the system, and as such, the interface design was a large portion of the project. It was required that the system was intuitive and easy to use by untrained users. The only way this would be possible is from an attractive, straightforward interface. This proved to be a challenge, however, since as functionalities increased, the system expanded considerably - making it hard to remain uncluttered.

4.2.1 Initial Prototyping

Prior to implementation, all designs went through a prototyping process to gain insight into what each page would be required to accomplish and ensure all features had been considered. Figure 4.2 illustrates the initial prototype developed for Event Tracker. A dashboard layout was envisioned, as at this point it was unclear on how many modules would be required for the system. A fluid dashboard would ensure that modules could be added in the future, with little changes to the design, satisfying requirement **#28**. Requirement **#29** states that the system had to be responsive and work well on smaller devices such as mobile phones, and so this layout would allow different modules to be stacked above each other on these screens. Paper prototypes were also created for mobile devices to fully understand the differences between these views.

Along with the larger wireframes for full page designs, each module also went through a prototyping phase. A subset of the developed wireframes for several of the modules is pictured in Figure 4.3. This process was extremely valuable and helped speed up implementation by having a constant, clear idea of how the final product should appear. After considering different interactions with the modules, it was determined that for any extra functionality a module may require, for instance, adding a user in the group management module or customising the set of categories shown to users, a pop-up modal would appear. This would ensure that the main

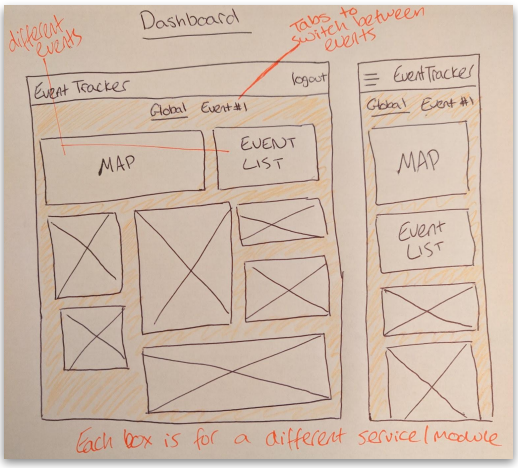


Figure 4.2: *Initial dashboard focused prototype, describing how Event Tracker will look on both large and small screens.*

design of the page would remain unchanged; a new page would not have to be loaded, and the modules could be kept minimalistic.

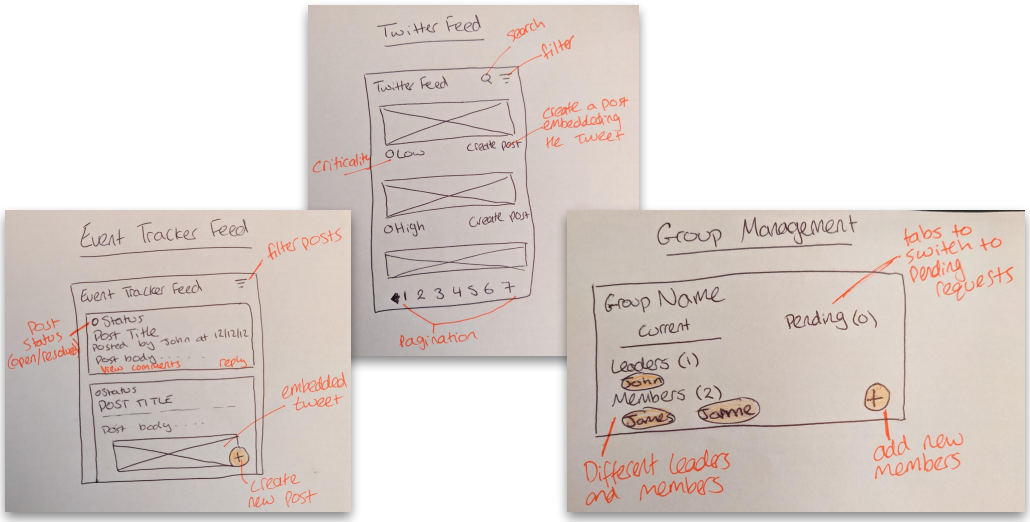


Figure 4.3: *Subset of the wireframes created for each module, highlighting different potential user interactions.*

4.2.2 User Dashboards

As shown in the initial prototypes, a dashboard design was chosen to contain the majority of Event Tracker's functionality to ensure the design was extensible while remaining attractive and easy to learn. Materialize was used as the frontend framework throughout the application, which ensured a modern, responsive design could be built with ease (Materialize 2019). Five different dashboards were built, namely *public*, *global*, *member*, *leader* and *admin*. These dashboards and associated modules discussed later, provide the base for meeting requirements **#2**, **#4**, **#10** and

#26. Appendices B.1 and B.2 picture full examples of a *global* and *leader* dashboard.

The *public* dashboard is available on the splash page of Event Tracker, allowing users to quickly view a subset of details regarding an event without requiring to login to the system. This aims to provide victims during a disaster with advice and guidance on how to remain safe. The *global* dashboard appears once logged in to the system, where similarly a subset of details can be found regarding all events being tracked. Modules are available on both dashboards allowing users to switch between events, either from a map or a list visualisation, satisfying requirement **#3**. The *member* and *leader* dashboards are then available once a user is a part of an organisation on Event Tracker, where they can then access further details about the event and gain communication functionalities. Finally, the *admin* dashboard grants system administrators higher priority functionalities regarding group management, discussed in further detail in Section 4.2.5.

4.2.3 Actionable Information Feeds

As discussed previously, getting key information to the right people quickly and efficiently is crucial for a successful crisis management system, and one of the main reasons why previous systems have failed. Event Tracker aims to combat this by generating multiple different actionable information feeds displayed on the different user dashboards. Each of the feeds is designed with a different intention of the value it can provide to the response agencies and volunteers, however, they all aim to improve the situational awareness of these users. Situational awareness is commonly referred to as the notion of knowing what is going on around you and developing a clear picture of the situation (Harrald and Jefferson 2007). A high level of situational awareness is vital in effective decision making in complex and dynamic environments, such as an ongoing disaster. However, a problem with this definition is that various individuals in different roles during a crisis may require a unique view of an event in order to respond effectively (Zade et al. 2018), and so improving the general situational awareness may not provide enough value. Hence, through Event Tracker, different actionable information feeds can be somewhat customised to allow users to focus on their specific needs.

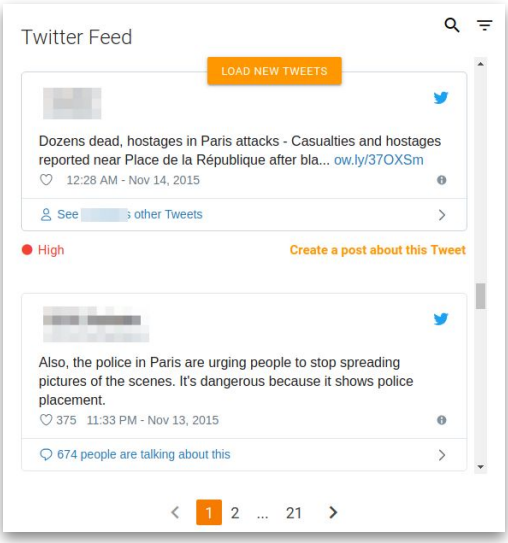


Figure 4.4: *The Twitter Feed, a real-time, filterable collection of all Tweets gathered for an event.*

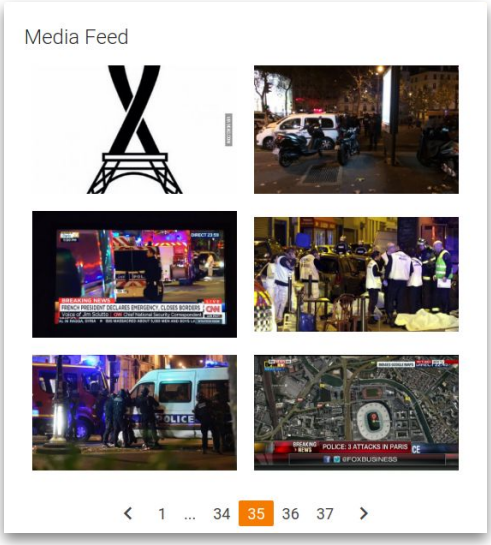


Figure 4.5: *The Media Feed, which extracts all media from the collection of Tweets.*

The *Twitter Feed*, pictured in Figure 4.4, is a real-time, filterable collection of all data gathered across the duration of a given crisis event. Tweets can be sorted based on either the date they were posted or by user importance and filtered based on their estimated criticality. The second actionable information feed to be developed was the *Media Feed*, shown in Figure 4.5. This feed extracts all imagery from the collection of gathered reports, potentially allowing response officers to assess damage during or after a disaster, or help volunteers identify locations where victims may require aid. Each image can be enlarged, at which point the original Tweet is also displayed in an overlay, to get more context about the image. Automatic content-based categorisation, through the TREC-IS initiative, examined in Section 2.2.1, is exploited by Event Tracker to provide emergency response agencies with a set of *Tweet Category Feeds*, to increase how quickly and accurately these agencies can navigate the collection of data to find actionable information. The categories displayed to users can be customised from a set of 25 different distinctive options such as 'Offical Reports' or 'Emerging Threats'. This feed successfully satisfies requirement **#14**, and due to categories such as 'Volunteer Requests', requirement **#25** is also met.

4.2.4 Criticality Alerting

Early warnings and alerts regarding critical information posted online during a crisis situation are imperative for a quick response from emergency services. Aid requests could be answered quickly, increasing, for example, the likelihood that a victim could be assisted in time. Event Tracker aims to accomplish this by again making use of the TREC-IS initiative to provide real-time identification of critical information. To successfully meet requirement **#15**, fast classification and text processing technologies are exploited to immediately label incoming messages with a criticality score of *low*, *medium*, *high*, or *critical*, which can then be displayed alongside Tweets on a user's dashboard. An example of this can be found in Figure 4.6, where two Tweets are displayed, both collected during the *2015 Paris Attacks* event. The first Tweet contains information regarding the ongoing hostage situation, including figures such as the current death count, hence has been marked as *critical*. The second Tweet simply consists of a user expressing sympathy over the attacks, and so has been labelled as *low*, as the content does not contain any crucial information which would be relevant to aid the response effort. This functionality, however, does not provide adequate notifications or alerts when new potentially critical information is collected, and so to combat this, and satisfy requirement **#17**, the *Critical Feed* was designed. This allows real-time highlighting of the reports which are predicted to contain the most critical and immediately actionable information, enabling a quicker response time from emergency services.

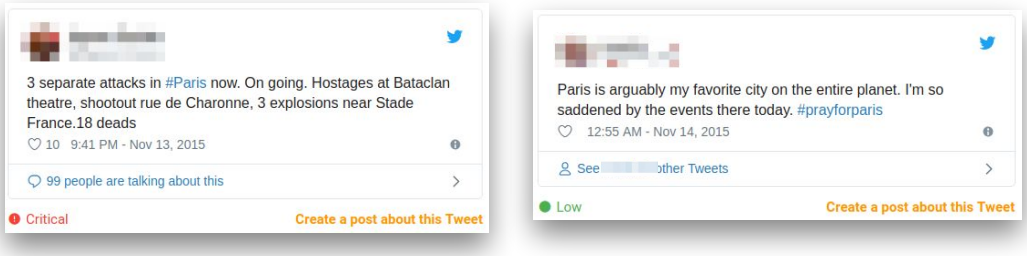


Figure 4.6: *Two example Tweets collected during the 2015 Paris Attacks event, displayed alongside their estimated priorities. The left is highlighted as 'critical', whereas the right is marked as 'low'.*

4.2.5 Group Management

To support crisis management, there are many associations that could help monitor and manage a disaster, from the Red Cross to smaller regional volunteer groups. Every event being tracked on the system can be related to multiple response associations, each of which may provide support in different manners. A large amount of the listed functional requirements centre around the aspect of group management, thus three main modules were designed to aid this. The first was the *Leader Group Management* module, pictured in Figure 4.7, available exclusively on the *leader* dashboard. Here, group leaders given the tools to manage the members of their group, satisfying requirements **#8** and **#9**. The second is the *Group Request* module (Figure 4.8), which allows users on Event Tracker to send a request to the group leaders to become a representative of their organisation. This module also allows leaders to request the creation of a new group to the administrators, required for **#12**. To meet requirement **#23**, an open volunteering group is set up for every new event, which any user is free to join through this module on the *global* dashboard. Lastly, the *Admin Group Management* module was designed to satisfy requirements **#5**, **#6** and **#7**, by allowing Event Tracker administrators to create different groups relating to any event and assign leaders to each. The module also allows administrators to accept any pending group requests created in the previous *Group Request* module.

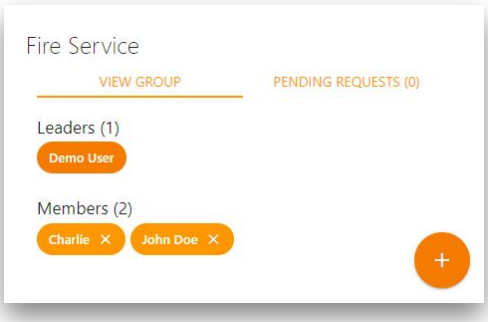


Figure 4.7: *Leader Group Management module, enabling leaders to view their group, add or remove members and accept pending member requests.*

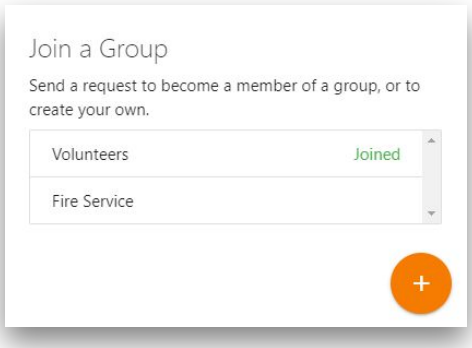


Figure 4.8: *Group Request module, allowing users to request to join a group, join the open volunteering group or request to create a new group.*

4.2.6 Communication Amongst Response Groups

To improve communication between the different response groups, the *Event Tracker Feed* was designed. These associations can communicate directly through this feed and its various supporting components, pictured in Figure 4.9, passing on any relevant information, satisfying requirement **#11**. As discussed, there has been a lack of successful systems that integrate formal response agencies with volunteers, and hence the open volunteering group can also access the *Event Tracker Feed*. The system can be used to increase the coordination and communication between these entities, with both volunteers on the ground reporting information, or digital volunteers from anywhere around the world highlighting critical reports to official response officers by embedding Tweets into their posts, serving requirement **#24**. This feed was also extended to increase the value of the system, by giving posts more features, such as a status, a due date and the option to make the post public on the *public* dashboard. This satisfies requirements **#13** and **#27** by allowing response officers to broadcast information to the public view of the

system, allowing victims to receive up-to-date guidance with instructions on how to proceed in their situation.

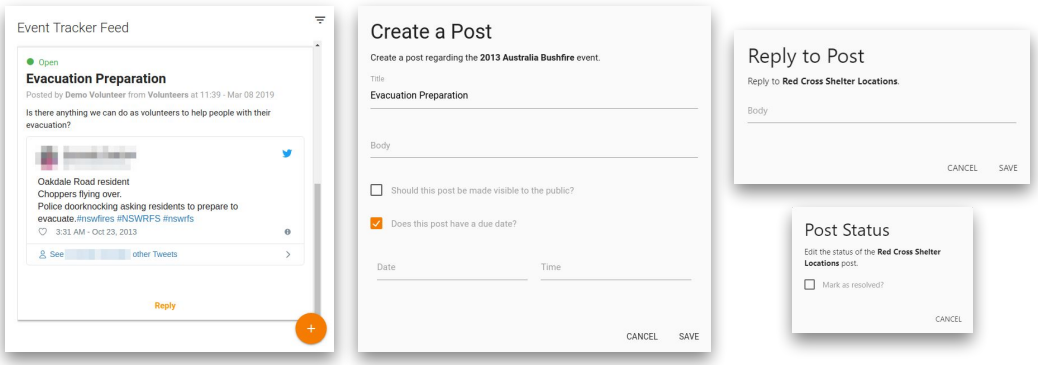


Figure 4.9: *Event Tracker Feed and associated components, enabling communication between response officers and volunteers on Event Tracker.*

4.2.7 On-request Information Access

Making use of the integrated search service described in Section 2.2.2, Event Tracker allows users to explore the collected data with ease by performing simple queries on the wealth of information, successfully satisfying requirement **#16**. Searches can be started from the *Twitter Feed*, where it was originally designed such that the *Twitter Feed* content would be replaced with the search results. It was then decided the feature may be more valuable to the users if multiple searches could be run concurrently. The design was changed such that each search creates a new module on the user's dashboard listing the results, allowing for multiple searches that can be easily added or removed. Both emergency response groups and volunteers could make use of this functionality to access specific information - for example, the Fire Service could run a search of 'fire'. While this service dramatically cuts down on the amount of information users may have to navigate, there is room for expansion, possibly by limiting searches by location or category.

4.2.8 Crisis Mapping

Requirement **#18** calls for crisis mapping tools, and as such the *Event Map* module was constructed. Figure 4.10 pictures an example of this module on Event Tracker during a simulation of the *2015 Paris Attacks* event, displaying both the locations of the event and the geo-tagged Tweets that have been collected over the duration of the crisis. Like the actionable information feeds, this aims to aid in increasing the situational awareness of the response agencies during disasters. By providing a different visualisation of the data, as opposed to limiting the dashboards to simple lists of reports, Event Tracker can become more appealing and interesting to the users.

4.2.9 User Authentication

A simple yet crucial requirement for the success of Event Tracker is requirement **#1**, which asks for users to have the ability to create an account and log into the system. This requirement did not involve much design, as only two simple static pages would be required, namely a sign in page and a sign up page. Both pages simply contain a form asking for users to fill in their details to gain further access to the system. It was decided that, as Event Tracker is prototype system,

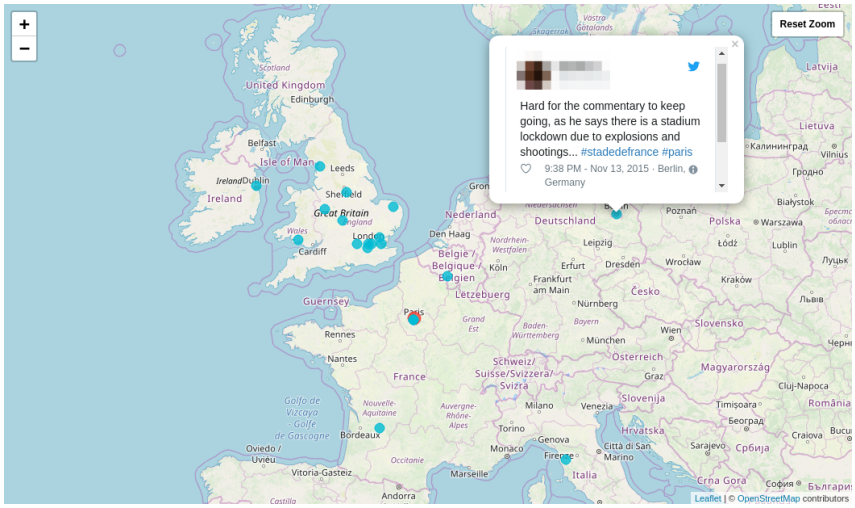


Figure 4.10: *Event Map for the 2015 Paris Attacks event, highlighting the locations of gathered reports and allowing these to be viewed in a pop-up.*

this should be as simple a process as possible, and as such common user authentication features such as email validation and password resetting were left out. The sign up page is pictured in Appendix B.4.

4.3 Tools and Technologies

Many tools and technologies could have been used throughout the development of Event Tracker. Whilst in some cases technologies to be used were restricted by the project requirements, others were not, and so multiple different options were considered.

4.3.1 Backend

The backend framework is the primary technology that was limited by project requirements. Requirement **#32**, states that, as per customer request, it should be possible to integrate Event Tracker with other components in the Emergency Public Information and Communication System (EPIC-Sys), a new initiative at the University of Glasgow designed to fill the technology gap in the emergency management sector. As such, the Play Framework must be used for the system to ensure each component is built using the same technology, increasing flexibility and the ease of which components can be integrated. Play Framework is built upon a lightweight, scalable architecture allowing for high-volume streams of data (Play Framework 2019). This makes the framework perfect for Event Tracker, where the high-volume, low-latency transfer of social media data is essential in successfully meeting project requirements.

4.3.2 Frontend

With the system relying on a large number of user interactions, JavaScript was an essential technology that would be heavily relied upon throughout implementation. Frontend technologies were not limited by requirements, resulting in three potential options for how JavaScript would be introduced. Firstly, vanilla JavaScript could have been leveraged, which would lead to much lower

overhead for the clients when loading the application. However, while this is a very attractive benefit, it would also increase development time considerably, as complex functions that come pre-packaged in frameworks are not available. Angular was considered as a second option, as it is a popular framework that has gained a large following. This framework offers a lot of functionality which can be implemented effortlessly, such as data binding and reusable components, however, these features come with a steep learning curve and a higher overhead compared to vanilla JavaScript (Angular 2019). The third option, Vue.js, provides a similar framework structure as Angular, with a slightly reduced feature set (Vue.js 2019a). With this reduction in features, however, the overhead is also much smaller than Angular, with the production distribution only taking up 30KB. For these reasons, it was decided that Vue.js would be used, as it offers a good compromise between vanilla JavaScript and Angular.

4.3.3 Databases

Data storage and transfer is an essential component of Event Tracker, and thus the choice of database was a major decision. As discussed in Section 4.1, two sets of data were to be stored, the entities such as events, users, and groups, and the collected social media data. Two types of database were considered for each set: relational and NoSQL. For the first set, a relational database was the best option, due to a large number of relationships between all the entities, which are much easier to implement compared to a NoSQL database. PostgreSQL, an open-source relational database management system, was chosen to store this data due to its popularity and wealth of documentation (PostgreSQL 2019). As a result of the second set of data containing a high-volume of potentially unstructured data, from both Tweets any services, NoSQL was the better option. Using MongoDB, a popular open-source document database (MongoDB 2019), it is ensured that low-latency data transmission is possible, crucial for the real-time criticality alerting on the system.

In addition, the Ebean library will be used to provide an object-relational mapping between the entities stored in the relational database (Ebean 2019). This library provides a simple yet performant means for reading and writing to the database, which will be a common occurrence within Event Tracker. Appendix A illustrates the final database schema for this data.

4.3.4 Message Broker

As discussed, the system was designed such that a message broker would be harnessed to handle the communication between Event Tracker and the classification service. RabbitMQ was chosen as the key technology to provide this functionality, due to its wealth of documentation and the ease and extensibility of which both the client and server can be configured (RabbitMQ 2019). Different producers, consumers, queues, exchanges can be quickly set up, allowing for a fully customised configuration to aid the transfer of data. The addition of an admin web interface also supports finding and resolving any issues within the data flow.

4.4 Summary

This chapter provided an overview of the Event Tracker architecture, before discussing the different components which make up Event Tracker, and how these were designed to meet the project's requirements. Tools and technologies were also examined, resulting in the Play Framework and Vue.js being used as the main technologies for the project, along with several other technologies for data storage and service interaction. Finally, the design process was examined, detailing the initial wireframes and how these led to the creation of the final deliverable.

5 Implementation

This chapter will review the implementation stage of Event Tracker. First, software engineering practices that were used will be examined. The integration of the backend services with then be discussed, followed by an in-depth examination of several key implementation details, challenges faced and how these were overcome. Finally, the different stages of the project's deployment process will be investigated.

5.1 Software Engineering Process

Throughout the project, several different software engineering processes and practices were introduced. These practices were each expected to aid the development process in different ways, which most accomplished. However, some did not add as much value as anticipated.

5.1.1 Version Control and Continuous Integration

A large focus was placed on the adequate use of version control throughout the project's development. GitHub was used to host the repository, ensuring that all code was backed up and would not be lost had any local development issues occur (GitHub 2019). The use of branching also allowed different features to be implemented separately, ensuring that if anything were to go wrong, a full working system would still always be present on the master branch.

Continuous integration was introduced to the project early on in development. Travis CI was used due to its good connection with GitHub, allowing pipelines to be triggered when code was merged into the master branch, indicating a feature was complete (Travis-CI 2019). The pipeline was set up to run both unit tests and a linting service, to ensure the correctness of the new code and that the style of the code met certain expectations. Unfortunately, this process did not perform as well as anticipated and did not work for a large portion of the project. It was decided that fixing this was not a priority, due to there only being a single developer on the project and so and instead both running tests and code linting would be completed manually before merging a new feature.

5.1.2 Issue Management and Time Tracking

The use of extensive issue management throughout all stages of Event Tracker's development ensured that it was always possible to find out exactly what had been done and what was left to do before the project was complete. As pictured in Figure 5.1, a Trello board (Trello 2019) was set up with different lists of issues, grouping tasks into categories such as 'Backlog', 'In Progress' and 'Done' to indicate their progress. This allowed for an efficient, easy to understand overview of the full project. Each issue was also given a label indicating what type of task it was, for example, 'Requirements Gathering', 'Service/Module' or 'Bug'. The TimeCamp integration with the Trello board also allowed time to be tracked for each issue. This was a big part of the project

management, to ensure time was being spent effectively. Not only did this grant the ability for issues to be looked back upon to see how much time had been spent, but for upcoming issues to be planned by estimating the time they were expected to take based on previous similar tasks.

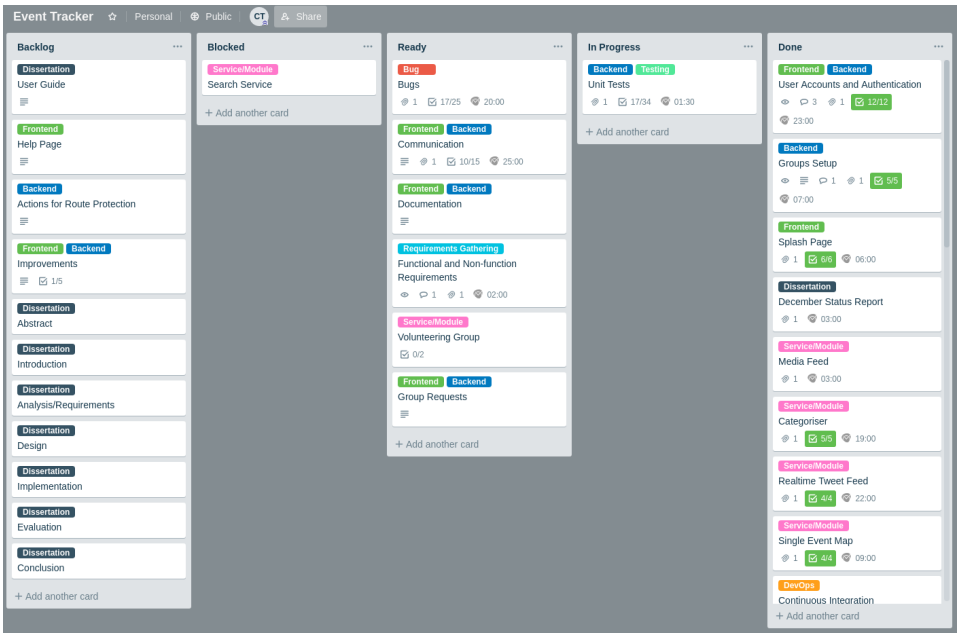


Figure 5.1: *Trello board for Event Tracker issue management. Issues are split into multiple lists to indicate their progress and labelled according to the type of task involved.*

5.1.3 Iterative Development

With the project being very easy to split into different sprints - either focusing on a certain type of user or focusing on the integration of a certain service - it was natural to follow an iterative development process. This also allowed requirements to be updated as the project developed and more functionality could be achieved through the introduction of different backend services. No strict time period was set for the duration of each iteration, however, as the project progressed, it became easier to predict how long each would take based on the time tracking discussed above. Extensive documentation was produced for each iteration, with weekly reports created identifying what was accomplished each week and outlining what was to be focused on the following week.

5.2 Service Interaction

Prior to the implementation of several of the Event Tracker features, the services which are harnessed by the system had to be integrated. Three main services were integrated with the platform - the background of these and a brief explanation of how they work can be found in Section 2.2. Each service was standalone and hence different interaction methods had to be employed to allow communication between each service and Event Tracker.

5.2.1 Stream Simulator

The first service to be integrated with the system was the stream simulator. This service is a version of the Social Media Crawler Streaming Client (SMCSC), however, it has been populated with past events rather than being connected with a live social media crawler. This was essential in order to effectively develop and evaluate the system as it would not be feasible to complete this only using live data. The stream simulator was harnessed by Event Tracker for two purposes:

1. Collect the details about all events which were available to be simulated.
2. Start an event simulation, allowing reports to be streamed into the system in pseudo-real-time.

On server startup, Event Tracker sends a request to the simulator, both retrieving its status and a list of events. These events are then be processed and stored as an entity in the relational database, ready for groups to be set up. A simulator page was set up on the system, found in Appendix B.3, where users can begin a simulation of any of these events through the controls pictured in Figure 5.2. A unique event identifier can be simply sent to the SMCSC which will begin streaming reports to the Event Tracker store at a speed specified by the user. A problem with this quickly arose as ongoing simulations could be cancelled midway if the user who began the simulation closed the browser. As the store is centralised for every user, this was not feasible. To solve this issue, this functionality was separated onto a persistent thread running on the web server, which successfully prevents this issue from occurring.

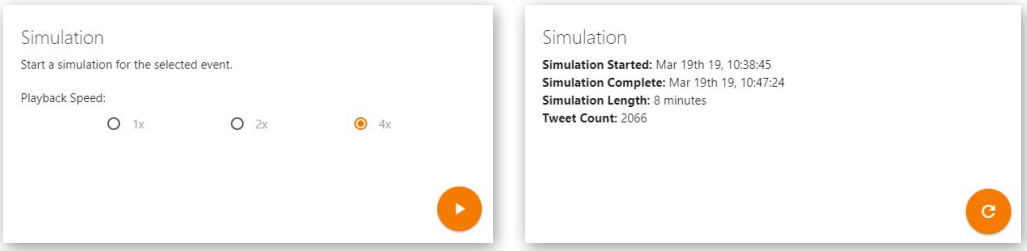


Figure 5.2: *Event simulator controls, allowing users to begin an event simulation at a desired speed and view the details of any ongoing simulation.*

5.2.2 Classification Service

The classification service was a crucial component for two key functionalities of Event Tracker, namely generating actionable information feeds and criticality alerting. As a large volume of reports would be required to be sent to and from Event Tracker and the service, along with the classification of each report taking an unknown length of time, it was decided to introduce a message broker to handle this interaction.

Figure 5.3 illustrates the data flow for this process. The code handling the SMCSC connection was modified so that incoming Tweets were not only sent to the store, they were also published to an exchange which would direct the Tweet into Queue 1. A consumer running on the classifier service then collects and processes these Tweets and then publishes the output back to the exchange, which directs the results into Queue 2. Finally, a consumer running on the Event Tracker web server processes the data from this queue and stores the Tweet categories alongside the Tweet itself in the store.

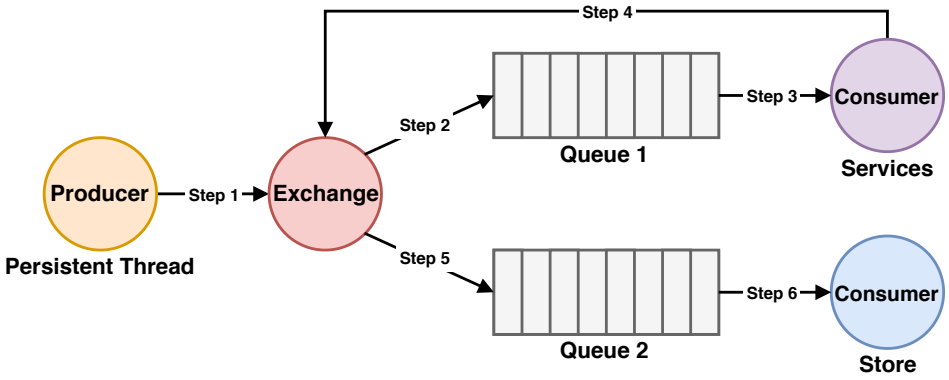


Figure 5.3: *Message broker data flow. Reports are sent from the persistent thread running on the web server to the classifier service. The service then returns any data to the store.*

5.2.3 Integrated Search Service

The integrated search service was the final service to be integrated with Event Tracker. As they are both components of the SUPER platform (SUPER 2017), the process to connect to this service was like that of the stream simulator. The process starts with a one-time request from Event Tracker containing a unique event identifier to register the related event on the service, which would then connect to the stream simulator to access and index the reports collected regarding the event, harnessing technologies provided by the Terrier Search Engine (Ounis et al. 2006). Once the data had successfully been indexed, a query (including the event identifier) could simply be sent to the service and a stream would be opened where the top results would be returned to be processed by Event Tracker.

Initially, whenever a new event was registered on the system (through use of the stream simulator), the request to register the event with the integrated search service would be sent. However, this led to resource complications for the service, as it was intensive to have all 29 simulated events indexed at once - a situation which would not occur during a real deployment of the system where it would be rare to have this many events occur at once. For this reason, it was decided that a new module would be added to the admin dashboard, effectively allowing these high-level users to 'start' and 'stop' and event, so that at any given time there would be few events registered with the service, allowing resources to be allocated effectively.

5.3 Real-time Data Streaming

The ability for Tweets to be displayed real-time on the user dashboards was crucial for both the *Twitter Feed* and the criticality alerting functionality. It would not be natural to assume users would repeatedly refresh the application to load new data and so to ensure emergency response officers were quickly notified by new data was imperative in the success of the application. The solution to this problem was to integrate WebSockets into Event Tracker, which enables a connection to be opened between the client and server, allowing for interactive communication (Navada 2018). Figure 5.4 describes this communication, where first, the connection between the client and server is initiated before the client sends a request asking for the Tweets relating to a specific event. At this point, the server will query the store and return the results, whilst also starting a thread which will send new messages containing any new Tweets as they are collected from the simulator. On receiving these Tweets, the client then checks to see the data type before filtering the reports and displaying them in the correct modules. This quickly highlighted an

issue with the information feeds, as when new Tweets were loaded in, the current data was pushed down. This meant that had a user been actively reading a report, it could be automatically moved away, leading to poor user experience. The solution to this was to add a 'load new Tweets', which would appear when new reports had been collected and would update the feeds with the new information on request from the user.

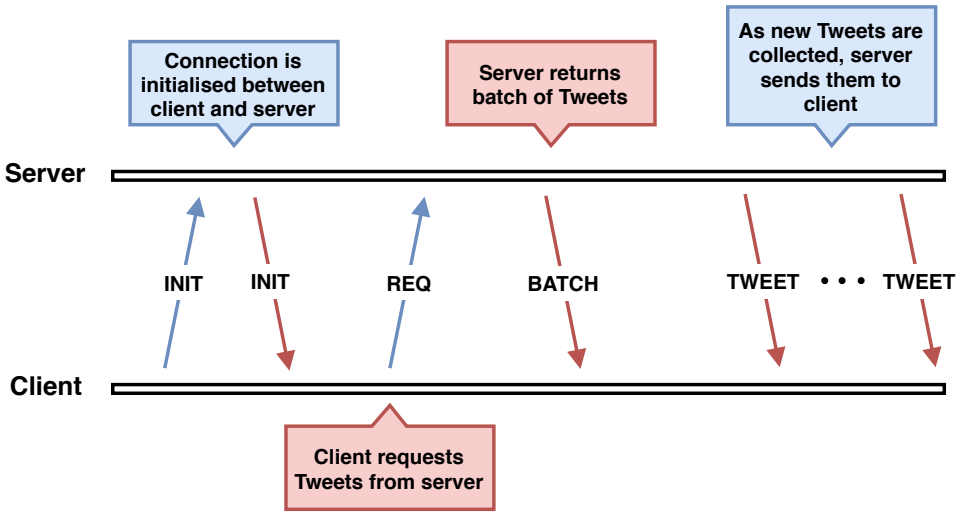


Figure 5.4: *WebSockets conversation between client and server to pass Tweets to the response agencies. Once a connection is initialised, the client sends a request for Tweets. The server then returns a batch of Tweets and begins a thread which will send new Tweets to the client as they are collected.*

Whilst the *Twitter Feed* and the *Critical Feed* were both successfully integrated with WebSockets to allow for real-time updating, issues were faced while setting up further components to allow for this functionality. In particular, difficulties were faced with the *Event Map* module, which was implemented using Leaflet (Leaflet 2019) and OpenStreetMap (OpenStreetMap 2019) and displays the locations of Tweets in a Geo-JSON overlay. Updating this overlay with the locations of incoming Tweets caused the whole overlay to reset, before showing again with the new data. This meant that whilst a simulation was running, the map would appear to flicker, which could be very distracting to users as they are focused on the system in a potentially high-stakes environment. In the end, this feature was disabled and due to time constraints, a better solution was unfortunately not found.

5.4 Information Rendering

The Twitter API had to be harnessed to display Tweets to comply with the display requirements outlined in Twitter (2019). This change highlighted a key issue with the performance of the *Twitter Feed*, whereby if all Tweets for an event were loaded into the feed, potentially hundreds of requests would be sent to the Twitter API at once, causing the system to slow down dramatically and sometimes refuse to send the requests. This solution to this problem was to implement pagination, splitting the feed into pages of up to 20 Tweets. This feature provided a good compromise between displaying as much data to the users with performance, whilst still complying with the required terms of service. The pagination controls were implemented as a reusable component, allowing the *Media Feed* to make use of them to solve similar performance problems.

A second issue with the *Media Feed* was caused since Event Tracker was developed and evaluated

using past event streams. As these contained reports dating back to 2011, it was found that there were many instances of media which led to broken links, causing a lot of thumbnails to not be displayed. To solve this, a simple check was set up on the server that would 'ping' each resource to determine if it still existed. Only images which passed this check would be passed down to the client, ensuring no broken links were present, and all thumbnails would render correctly.

5.5 Server-side Rendering

Originally, Event Tracker was implemented such that there were two individual projects - a Vue.js frontend running on one port, and a Play Framework backend running on another. This was to further separate concerns and increase flexibility. However, this led to issues primarily regarding user authentication, as it was difficult to ensure there was a secure connection between both instances and it was an uncommon setup leading to limited documentation. The *sbt-vuefy* library (GIVE.asia 2019) allowed Vue's single file components to be integrated with Play to allow the entire application to be served as a single project with all components rendered server-side. This proved to be a viable solution in securing the user authentication data flow and also improving search engine optimisation - although this was not a requirement of the project. This design change, however, ultimately led to sub-optimal unit testing, as it was no longer possible to adequately test the Vue components. This bug was highlighted to the developer of the library, who stated that it was indeed a problem with the package and there was no immediate solution. As such, little could be done to rectify the situation as the bug was found well into development. This indicates a lack of planning before the design decision was made and a flaw in how often tests were being written. In future, a test-driven development approach (Koutifaris 2018) could have highlighted this problem immediately and allowed another method to be taken.

5.6 Frontend

As the majority of the project focused on creating efficient and user-friendly modules to display the different information feeds, it was critical that processes were in place to achieve this, whilst still allowing modules to be created quickly and methodologically. Two main practices were found to be of great use whilst implementing the frontend of Event Tracker: the use of both components and centralised state management.

5.6.1 Components

Using Vue.js, reusable components could be developed with ease, each of which contains its own styling and logic, separating concerns. This was essential in building Event Tracker to match the wireframes developed earlier in the project, as multiple different user dashboards had to be created with a varied assortment of modules in each. Each module being built as an individual component allowed a dashboard to be easily built using any selection of desired modules. Components were also built to be reused even within these modules - in particular, loading spinners, pagination controls and displaying Tweets. Components could be implemented to take in properties when being added to a page, which could, for example, turn specific features on or off. This was used in places such as the Event Tracker Feed, where the ability to create posts could be easily turned off on the public and global dashboards, without requiring a full new module to be implemented.

5.6.2 State Management

To ensure that requirement **#28** was met, which asks for Event Tracker to be modular, a good system for state management was essential. Vuex was used to provide this functionality due to its close integration with Vue.js, allowing a global state management store to be set up with ease (Vue.js 2019b). Rather than each component containing its own state and interacting directly with the web server, which may be a more classical solution, each Event Tracker module first passes through a central store. Components can request actions to be completed to the store, which passes these requests to the web server. Once the action is complete, rather than only sending the response back to the calling component, the central state is updated, ensuring that all modules will be updated with the new information if necessary. This enables all data to be consistent throughout the web application, without requiring manual updates. This process has many benefits, including both ease of development and more efficient storage of data on the client-side - key for better performance when high volumes of data are required. This store also aided in the creation of the modular components which could be added to any page without worries as data was available site-wide.

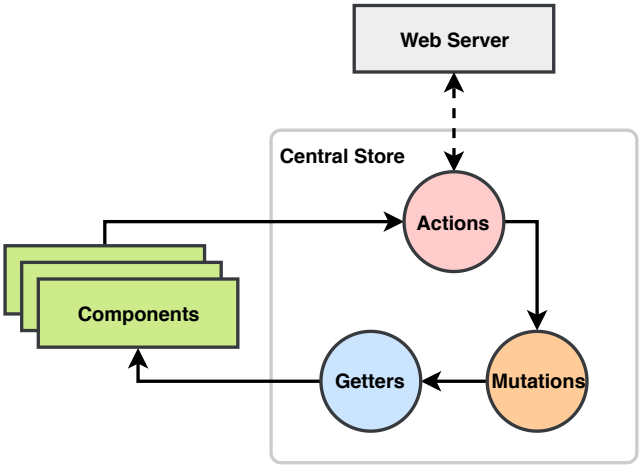


Figure 5.5: *Centralised state management data flow. All data is passed through a central store in which all components read from, ensuring data is consistent across the application. Figure adapted from Vue.js (2019b).*

5.7 Deployment

An early deployment of Event Tracker midway through the project was planned to demonstrate several functionalities of the system. This deployment highlighted major difficulties in deploying the system, where issues arose regarding creating a production version of the system, along with getting both databases set up correctly. It was clear that the current set up at the time was not feasible and so in order to improve this Docker (Docker 2019) was used. Docker, and specifically Docker Compose (Docker 2018), enabled individual containers to be set up, one for the web server and one for each database. These containers could simply be started on any machine, successfully deploying the system.

Towards the end of the project and a final deployment was being planned, it was decided that as the container setup was effective in deploying Event Tracker and its associated data stores, that the backend services should also be included in this setup. This would allow a full working

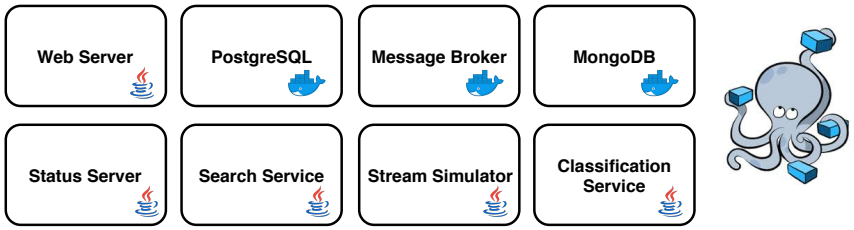


Figure 5.6: *Docker Compose multi-container overview for deploying Event Tracker. Eight containers were necessary for deploying the system, either built from a JAR distribution or from a pre-defined Docker image.*

system to be deployed on any machine with ease without needing to consider if services could be accessed through the University firewall, for example. Compose was again used, however, eight containers were now required increasing the system complexity. As shown in Figure 5.6, pre-defined Docker images could simply be created for both data stores along with the message broker. For the remaining containers, JAR distributions were created for the web server, each service as well as a new status server to monitor the status of each service. All these containers could easily communicate with one another over an internal network.

5.8 Summary

This chapter provided a review of the implementation stage of the project. Software engineering practices were examined, highlighting the key successes and downfalls of each. An explanation was provided on how the existing backend services were integrated with Event Tracker, followed by a discussion of several key implementation details, along with any challenges which arose throughout development. Frontend management was reviewed, explaining the benefits of reusable components and centralised state management, before finally looking back on the deployment process of Event Tracker.

6 Evaluation

This chapter details the evaluation methods which were used for the project. These include unit testing, performance evaluation and a significant user evaluation. The results of each approach will be examined, to effectively evaluate the Event Tracker system. Finally, the project requirements will be examined to validate that they were met by the end of the project.

6.1 Unit Testing

Unit tests were created to ensure the correctness of the code behind Event Tracker and that the system performed as expected. Tests were created for a large majority of the system; however, it was decided that several components could not be tested effectively. These components handled the connection and integration with the existing backend services, therefore to test this code, it would be required to mock each service. The decision was made to forgo this, as it could mean a lot of new code would be required - to the point that it would not be worth testing the components as the code would have changed significantly. The JaCoCo library (JaCoCo 2019) was used to create a code coverage report, pictured in Figure 6.1. The tests achieve a line coverage of 71% and a branch coverage of only 37% - whilst this may be low, it is also worth noting that the use of libraries such as Ebean, used for object-relational mappings between entities (Ebean 2019), introduced a lot of auto-generated code into the project which was found to be very difficult to test.

Jacoco Coverage Report

Element	Missed Lines	Total Lines	Cov.	Missed Branches	Cov.	Missed	Methods	Missed	Classes
router		662	63%		29%	22	88	0	3
store.repositories		89	1%		0%	15	16	0	1
controllers		217	85%		61%	9	37	1	7
views.html.dashboards		21	0%		n/a	10	10	2	2
views.html		150	88%		66%	11	26	0	5
models		296	94%		40%	83	468	0	12
store.models		11	9%		n/a	4	5	0	1
service		39	76%		62%	8	20	0	5
providers		53	88%		78%	5	27	0	6
security		15	66%		0%	3	8	0	3
exceptions		8	75%		n/a	1	4	1	4
module		9	100%		n/a	0	2	0	1
Total	448 of 1,570	1,570	71%	623 of 1,002	37%	171	711	4	50

Figure 6.1: *JaCoCo code coverage report indicating a line coverage of 71% and a branch coverage of 37%.*

A major limitation to the unit testing was found when attempting to test the frontend code. As described in Section 5.5, a design decision was made which merged the frontend and backend into a single codebase. After consulting the creator of the package that was used to accomplish this, it was found that it was not possible to add unit tests for the frontend code. Unfortunately, this meant that the only way to test the user interface and interactions was manually and through the user evaluation, which is a sub-optimal solution. Upon reflection, further research should have been completed before merging the codebase to avoid this situation, as the drawbacks were only realised once it was too late to feasibly separate the code again.

6.2 Performance Evaluation

Non-functional requirement **#32** states that the system must perform well with quick load and response times. In order to determine whether this requirement has been successfully met a performance evaluation was carried out making use of Google's auditing tool, Lighthouse (Google 2018). This tool provides metrics regarding website *performance*, *best practices* and *accessibility*, along with suggestions on how to improve these aspects of the website. Table 6.1 displays the results from running these audits on each of the pages on Event Tracker. Audits were carried out on three different machines and networks, with an average value being calculated for each score. Along with the *performance*, *accessibility* and *best practices* scores which are marked out of 100, the time to *first contentful paint*, the *time to interactive* and the *estimated input latency* were also measured.

Table 6.1: *Performance evaluation results. Performance, accessibility and best practices are scored out of 100. Performance was evaluated on three individual machines on different networks and an average score was calculated.*

Page	Performance	Accessibility	Best Practices	First Contentful Paint (s)	Time to Interactive (s)	Estimated Input Latency (ms)
Public dashboard	86	83	65	0.3	5.3	220
User dashboard	78	92	66	0.65	7.9	255
Simulator	100	89	72	0.2	0.4	20
Log in	99	91	80	0.3	1.5	20
Sign up	100	91	80	0.2	0.35	20

The results show that Event Tracker performs well in most cases. Pages with much more content such as the dashboards perform worse than the simple pages such as the simulator controls or registration pages, however, they still scored fairly well in most categories. It is clear that there is room for improvement regarding the *time to interactive* and *estimated input latency* scores on the dashboards, however, upon further analysis, it was found that these scores were to the detriment of the Tweet rendering. Despite adding pagination to combat this, as discussed in Section 5.4, many Tweets are still rendered on page load, leading to delays. Suggestions to improve these scores primarily related to an excessive DOM size, however, with such a content-heavy application, this is hard to avoid. As such, it was decided that as the system still received high scores throughout the audits, requirement **#32** was satisfied.

6.3 User Evaluation

As Event Tracker is designed to be a public system used by users around the world, a user evaluation was necessary in order to fully examine how effective the system is in achieving its goals. The evaluation conformed with all requirements in the School of Computing Science ethics checklist, which can be found in Appendix C.

6.3.1 Experiment Methodology

Two main experiments were developed in order to effectively evaluate Event Tracker. The experiments aimed to gather results regarding the functionality presented by Event Tracker – the first to ensure everything worked as users would expect and the second to reach a conclusion on

how effective certain features were, to determine if the system could indeed benefit emergency response agencies. Following these experiments, a questionnaire was also filled in by participants to elicit both quantitative and qualitative feedback.

Experiment 1

For the first experiment, participants will take part in completing two tasks, both designed around specific use cases for Event Tracker. Tasks were developed such that in order to be completed successfully, participants would require to make use of many different features. These tasks can be found in Appendix D.1. Ten participants, who had never seen the system before, were chosen to take part in this experiment. Participants were then split into two groups. Participants in the first group would receive training in how to use Event Tracker prior to attempting the tasks; participants in the second group would not. The experiment is designed this way to not only test the functionality of Event Tracker, but to gain insight into whether training would be required by emergency services if the system were to be deployed in the future. Ideally, both sets of participants would perform equally, indicating that the design of Event Tracker was successfully easy-to-use, and thus training time would not be necessary. Training time can be correlated with the money and effort response agencies would be required to spend on personnel training, and as such, a lower training time suggests less money and effort required. The following metrics will be measured in order to reach a conclusion:

- **Time on task**, which is the duration of time taken from starting the first step to completing the final step of a task.
- **Non-critical errors**, which are errors which occur throughout the completion of a task, but do not impact the task completion. For example, a participant going through unnecessary steps to complete a task rather than following the optimal path.
- **Critical errors**, which are errors which occur throughout the completion of a task that causes some deviation from the set task and results in an incorrect outcome. For example, a participant starting a simulation for the *2013 Australia Bushfire* event, when the task stated to start a simulation for the *2013 Boston Bombings* event. For the purposes of this evaluation, participants will be allowed to continue after making these mistakes to complete the rest of the task, if possible.

Experiment 2

Following the completion of the first experiment, participants would then be asked to partake in a second experiment. Again, the experiment consisted of two tasks, which can be found in Appendix D.2. For each task, participants will again be split into two groups, the first of which will have all developed features available to complete the task, whilst the second group have certain features disabled. The first task asks participants to find as many **critical priority** reports as they can find in four minutes relating to the *2018 Florida School Shooting* event. Whilst the group with features disabled will still be able to see estimated Tweet criticalities on the *Twitter Feed*, *Category Feeds* and from running searches, they will not have the *Critical Feed* or Tweet filtering enabled. The second asks participants to find as many **emerging threats** as they can in five minutes regarding the *2013 Australia Bushfire* event. This time, participants with features disabled will not have access to the different *Tweet Category Feeds*. Participants with this feature will be expected to customise these feeds to view the 'Emerging Threats' category. This experiment enables the inclusion of these features to be evaluated to determine if they have the potential to provide value to emergency response agencies and volunteers. Ideally, participants with all features enabled will perform much better than those without, signifying that the use of Event Tracker will enable end-users to identify and respond to important information quickly, aiding the overall response effort during a disaster or crisis situation. The following metrics will be measured for this experiment:

- **Time on task**, which is the duration of time taken for participants to correctly find all expected Tweets. Due to the time constraints applied to these tasks, the task will be stopped once the time on task reaches the specified value.

- **Tweets Found**, which is the number of reports found by the participant to signify either a critical Tweet or an emerging threat, depending on the task.
- **Correct Tweets Found**, which is the number of reports marked by the participant that is expected. For the first task this means the number of reports found that are actually marked as critical; for the second task, this means the number of reports that the classifier has also marked as emerging threats.

Post-evaluation Questionnaire
Finally, once participants have completed all four tasks across the two experiments, they will be asked to fill in a post-evaluation questionnaire, which can be found in Appendix E. The questionnaire consists of questions designed to elicit both qualitative and quantitative feedback. The quantitative section is based on the Questionnaire for User Interaction Satisfaction (QUIS), a standard method commonly used to assess the user's subjective satisfaction with a system. The QUIS is separated into several specific aspects to be evaluated, including overall reaction, screen factors, terminology and system feedback, learning factors and system capabilities (Harper and Norman 1993). The qualitative section contains open-ended questions, such as asking for any suggestions on how the system could be improved and whether the participants believe Event Tracker could be a successful emergency management tool and their reasoning behind this.

6.3.2 Pilot Study

Prior to the full user evaluation taking place, a pilot study will be carried out, where a single participant will attempt to complete the set evaluation tasks. The aim of this process is to ensure that the tasks were suitable and pick up on any remaining bugs that may be fixed before the main evaluation. This process also allowed insight to be gained regarding how long each evaluation would take to complete, to ensure it would be feasible to get participants to perform the experiments. Several issues were noted whilst carrying out the pilot study - several points about user experience improvements, but also in the tasks themselves. User experience issues brought up consisted of a lack in informing the user what the system was doing and the list of events appearing in a somewhat random order, making it difficult to find specific events in the *Current Event* feed. This led to notifications being added throughout the website which appear when certain actions occur such as starting a simulation and the list of events being sorted alphabetically. Another issue was that the tasks had originally been written out as a paragraph, which was found to make it difficult for the participant to keep track of where they were and so this resulted in changing the format into a numbered list, which was more desirable and easier to follow. Finally, it was clear that the evaluation required a significant amount of time per participant - well over an hour. Originally, three tasks had been assigned to both experiments, so one task from each was removed, with the remaining tasks tweaked slightly to ensure the functionality being tested in the removed tasks could still be evaluated. This decreased the evaluation duration to around 40 minutes per participant, which is still a considerable time investment, but due to the amount of functionality to be evaluated, it was determined to be necessary.

6.3.3 Results

Many conclusions can be drawn from the results of the user evaluation. The two experiments allowed many different aspects of Event Tracker to be tested and the post-evaluation questionnaire obtained a good overview on what participants liked about the system and highlighted key areas that require improvements before a real deployment could take place.

Experiment 1
The goal of the first experiment was to evaluate how necessary training would be before emergency services could make full use of the system during a real crisis situation. Two tasks were

completed by ten participants successfully. The results from the first task are displayed in Table 6.2, where, on average, the trained participant's time on task was 3 minutes and 56 seconds, whereas the untrained participants completed the tasks on average in 4 minutes and 38 seconds - 42 seconds slower. The two fastest participants were trained, however, the following three were untrained and completed the scenario quicker than the remaining trained participants. Untrained participants made 7 total errors across all participants, while their trained counterpart only made 5.

Table 6.2: *Experiment 1 task 1 results.*

Participant	Trained?	Time on Task	Non-critical Errors	Critical Errors
A	Yes	2:37	1	0
B	No	6:43	1	0
C	Yes	4:11	1	0
D	No	6:38	2	0
E	Yes	6:06	0	1
F	No	3:38	2	1
G	Yes	2:15	0	1
H	No	2:44	0	0
I	Yes	4:30	0	1
J	No	3:28	0	1

The second task generally took much longer for participants to complete than the first - however, the task had more steps and a higher complexity in order to test more of the functionality. As shown in Table 6.3, for this task, the average time on task for untrained users was a more significant 3 minutes and 11 seconds slower, with all trained users finishing quicker than all untrained users. Trained participants also made half as many non-critical and critical errors whilst completing the task than the untrained participants.

Table 6.3: *Experiment 1 task 2 results*

Participant	Trained?	Time on Task	Non-critical Errors	Critical Errors
A	Yes	5:42	3	0
B	No	9:55	3	0
C	Yes	4:41	0	0
D	No	6:35	1	0
E	Yes	5:24	1	1
F	No	7:25	2	1
G	Yes	4:00	0	0
H	No	9:11	3	0
I	Yes	6:02	1	0
J	No	8:45	1	1

While it is clear the trained participants performed on average better than the untrained participants in both tasks, this was to be expected. If this were not the case, the training would have somehow made participants perform worse. The results from the first task did not show a significant difference between the two sets of participants, while the second task resulted in a larger deviation between the two groups. However, the untrained participants generally encountered the same problems which led to their demise, highlighting several user experience issues. These include hard to find buttons to perform tasks such as starting a search or customising the *Tweet Category Feed* categories, and lack of instruction when adding members to a group. Both trained

and untrained users struggled when switching to different event dashboards - potentially due to the global dashboard appearing similar to the user's specific event dashboard, causing users to not realise they were on the wrong page. A simple solution to this would be to remove some of the modules from the global dashboard to make it clear that users should only use this page to quickly gain an overview on the different world events, rather than using the page for event monitoring.

Comparing the results indicates that although training improved performance, it was not strictly necessary in order to perform well. A potential compromise which could improve performance whilst decreasing the time or money required to train emergency response officers would be to include an interactive walkthrough on Event Tracker. This would step users through all the different features explaining each one and how they can be used. Although a help section already exists on the system, as its use was not mandatory, participants were reluctant to use this and preferred taking a trial-and-error approach to the tasks.

Experiment 2

The second experiment aimed to evaluate how effective certain features of Event Tracker are, namely the Tweet filtering, the *Critical Feed* and the *Tweet Category Feeds*. Again, all ten participants completed the two tasks. Table 6.4 displays the results from the first task, where participants who still had access to Tweet filtering and the *Critical Feed* managed to correctly find critical priority Tweets much more efficiently than the participants lacking these features. Almost all participants with the features enabled found all 5 critical Tweets in under 2 minutes, with only one participant only managing to find 3 as they failed to make adequate use of the modules available. No participants with the removed features managed to correctly find all critical Tweets within the time constraints - however, the closest managed to find 4 of the 5. These results clearly show that the Tweet filtering and *Critical Feed* functionalities are effective in vastly decreasing the time it takes to identify potentially important information during a disaster.

Table 6.4: *Experiment 2 task 1 results.*

Participant	Feature Removed?	Time on Task	Tweets Found	Correct Tweets
A	Yes	4:00	4	4
B	No	1:05	5	5
C	Yes	4:00	3	3
D	No	1:38	5	5
E	Yes	4:00	1	1
F	No	4:00	3	3
G	Yes	4:00	2	2
H	No	1:10	5	5
I	Yes	4:00	2	2
J	No	1:01	5	5

Further conclusions can be drawn from the second task. Again, participants with the feature enabled did far better than those without. Although only three participants correctly identified all expected emerging threats, the remaining participants without removed features were on track to find all the Tweets - but ran out of time. These participants spent too long trying to find emerging threats with different modules before finding the emerging threats category which contained all the correct Tweets. Participants without this feature performed poorly, with the highest number of correct Tweets being found being only 3 out of the possible 20. This shows just how effective the categorisation of reports is and the benefit it could bring during a disaster to aid the response effort. The number of Tweets to be navigated through is cut down immensely, allowing for crucial reports to be identified and acted upon quickly.

An important observation which can be made following the results of the second task is that

Table 6.5: *Experiment 2 task 2 results.*

Participant	Feature Removed?	Time on Task	Tweets Found	Correct Tweets
A	Yes	5:00	15	12
B	No	5:00	6	0
C	Yes	5:00	10	2
D	No	5:00	8	2
E	Yes	4:39	23	20
F	No	5:00	5	0
G	Yes	3:37	20	20
H	No	5:00	10	2
I	Yes	4:21	20	20
J	No	5:00	10	3

whilst the use of Tweet categorisation significantly improved the results, it is hard to determine what a 'correct' Tweet was. For this evaluation, Tweets were regarded as correct if they had been classified as an emerging threat, however, multiple participants highlighted the same several Tweets as an emerging threat which were marked as incorrect. This indicates that the classifier may not be highly accurate and these reports did, in fact, contain the details of potential emerging threats. Two main conclusions can be made following this observation. The first is that the classifier service should perhaps be changed to use a different technology such as AIDR, described in Section 2.1.3, to improve the accuracy of the classification. The second conclusion is that this behaviour indicates that a crowd-sourcing component could be introduced to Event Tracker, which would allow users to manually tag reports as specific categories, which could potentially vastly improve the categorisation effort.

Post-evaluation Questionnaire

Analysis of the post-evaluation questionnaire results revealed that, for the most part, users thought Event Tracker is fairly satisfying and stimulating to use. Almost all users believe the system is flexible, with only one participant disagreeing. Participants agreed that characters on the screen were easy to read and both highlighting on the screen and screen layouts were helpful, however, mixed feedback was received regarding the amount and arrangement of information on the screen. Whilst most participants stated that an adequate amount of information was displayed in a logical manner, several participants disagreed, giving neutral-low scores. Participants unanimously agreed that terminology throughout Event Tracker was consistent and precise, but several thought that instructions for commands or correcting errors could have been clearer. It was found that the system almost always kept the participants informed about what it is doing, and performing an operation generally led to a predictable result. The most varied response elicited by the questionnaire regarded how easy it was to learn advanced features, where several users thought it was very difficult, yet several thought it was very easy. After further analysis, it was found the users who thought it was difficult tended to be the participants who did not receive training before the first experiment - indicating that the use of either training or an interactive walkthrough could have improved this score. Again, users unanimously agreed that tasks could almost always be performed in a straight-forward manner and that the steps to complete a task followed a logical sequence. It was determined that the system speed, response times and the rate at which information is displayed were fast enough, and most users did not feel like they were left waiting as pages load.

The qualitative feedback section elicited several suggestions for Event Tracker, however, many of these were user experience related and had been picked up earlier in the evaluation. Reoccurring suggestions asked for it to be easier when adding group members; a more obvious search bar; making it easier to customise categories, and making the different dashboards more obvious. All

bar one participant believed Event Tracker could be a useful tool for both emergency response agencies and volunteers during a disaster, with a single participant stating that they think just using Twitter's own search functionality would be enough. Despite this, the results from the second experiment disagree, since it was found that criticality estimation and Tweet categorisation were effective in identifying potentially crucial reports. Several participants did not know if the system could be a useful tool in providing victims during a disaster advice and guidance on the steps they should take. This can be primarily related to a limitation in the user evaluation, as tasks were removed to shorten the evaluation, as mentioned previously. This, unfortunately, meant that there were no tasks dedicated where participants acted as a victim requiring help, leading to this functionality not being fully evaluated.

6.4 Requirement Validation

The final stage of the evaluation process was to determine if the functional and non-functional requirements outlined in Section 3 had successfully been met or not. As discussed when these requirements were stated, several requirements were added throughout the development of the project, and so only the final set of requirements will be validated. As they are defined as out of scope for the project, any *Won't Have This Time* requirements will not be included in this evaluation.

Overall, 30 of the 32 *Must Have*, *Should Have* and *Could Have* requirements were met by the end of the project - approximately 94%. All 20 *Must Have* requirements were met; all 8 *Should Have* requirements were met; however, only 1 of the 4 *Could Have* requirements were met. The remaining 3 requirements that were not met were requirements **#19**, **#20** and **#21**, which asked for push notifications regarding new posts or group updates; the generation of event summaries, and the creation of an event timeline, respectively. These requirements were not met due to time constraints of the time project, however, would provide extra value to the system and should be considered for future work. Appendix F provides reasoning as to why each requirement is believed to have been met.

6.5 Summary

This chapter detailed the assortment of evaluation methods used for Event Tracker. First, unit testing was looked at, where it was found that the tests were not as complete as they could have been due to limitations in the project design and poor planning. Next, the performance of Event Tracker was measured using metrics such as *time to interactive* and *estimated input latency*, where it was found that whilst there may be room for improvement, Event Tracker performs well for its purpose. An extensive user evaluation also took place, where several conclusions were drawn regarding the effectiveness of the Event Tracker functionality, and various user experience issues were highlighted. Finally, upon examination of the project's functional and non-functional requirements, it was validated that 94% of the requirements were met.

7 Demonstration Paper

This chapter will briefly describe the demonstration paper about Event Tracker submitted to SIGIR 2019. The paper can be found in full in Appendix G.

7.1 SIGIR 2019

The Special Interest Group on Information Retrieval (SIGIR) is an organisation which promote the research and development of information access technologies (SIGIR 2019). A demonstration paper regarding Event Tracker has been submitted to SIGIR 2019, the 42nd International ACM SIGIR Conference. The paper, titled *Event Tracker: A Text Analytics Platform for Use During Disasters*, discusses the motivation behind the development of Event Tracker, followed by an examination of related works, before investigating the system itself. Several key functionalities are highlighted, namely the generation of actionable information feeds, criticality alerting and the communication amongst response associations. The paper's abstract is as follows:

> Emergency management organisations currently rely on a wide range of disparate tools and technologies to support the monitoring and management of events during crisis situations. This has a number of disadvantages, in terms of training time for new staff members, reliance on external services, and a lack of integration (and hence poor transfer of information) between those services. On the other hand, Event Tracker is a new solution that aims to provide a unified view of an event, integrating information from emergency response officers, the public (via social media) and also volunteers from around the world. In particular, Event Tracker provides a series of novel functionalities to realise this unified view of the event, namely: real-time identification of critical information, automatic grouping of content by the information needs of response officers, as well as real-time volunteers management and communication. This is supported by an efficient and scalable back-end infrastructure designed to ingest and process high-volumes of real-time streaming data with low latency.

7.2 Summary

This chapter shortly discussed the demonstration paper about Event Tracker submitted to SIGIR 2019. The paper's abstract was also given, providing a short summarisation of the paper. The full paper is attached in Appendix G.

8 Conclusion

This paper so far has discussed the motivations and aims behind Event Tracker, followed by the requirements set to accomplish these. The design and implementation of the system were analysed before an extensive evaluation was carried out. This final chapter provides a summary of the whole project, before looking at any future work that could be carried out to further increase the value of Event Tracker. Finally, a reflection will be given regarding the project and the processes exploited throughout.

8.1 Summary

In summary, Event Tracker is a modular and extensible prototype system designed to support the monitoring and management of events during crisis situations. The system leverages a flexible architecture, which is designed for low latency and high-volume streams of data to provide functionalities such as the generation of actionable information feeds and criticality alerting. Event Tracker also enables the communication between various emergency response agencies and volunteers and provides additional functionalities such as crisis mapping and on-request information access. The system was evaluated in a variety of ways, including unit testing, performance evaluation, user evaluation and requirement validation. It was found that with its different functionalities, Event Tracker could prove highly effective in supporting the response efforts of both emergency response agencies and volunteers and aid in bridging the gap between these two entities. The system has proved to be highly usable and learnable, which could reduce emergency response agency personnel training costs during a real deployment of the system. A live demonstration of Event Tracker can be found at `http://eventtracker.charliethomas.co`.

8.2 Future Work

There lies potential in expanding the Event Tracker platform in both improving issues brought up in the evaluation stage and by increasing the functionality through the integration of additional backend services. Evaluating the system with real users identified several key user experience problems with the system. Whilst it was found that the system was quick to learn, to increase overall system usability and ensure that the costs of training personnel are kept low, it would be essential to fix or improve these problems prior to a real deployment. An interactive walkthrough was highlighted as a key feature which could help to ensure users knew exactly how to navigate the website and how each module worked. Multiple smaller design issues were also flagged, such as hard to find buttons and difficulties when completing group management tasks. While individually these are not big issues, together it is clear the system could be better designed.

A second large takeaway from the user evaluation is that of crowd-sourcing. Several participants highlighted the same Tweets as 'emerging threats' that the classifier did not - indicating that the classifier's accuracy may not be high enough, and important information could be missed by

response officers. In addition to the automatic content-based classification, the ability for users to manually tag reports as certain categories could potentially increase the added-value of the *Tweet Category Feeds* by ensuring less data can be missed.

Due to the flexible architecture Event Tracker is built upon, it should be a trivial process to increase the functionality and hence the value of the system by integrating more backend services, in addition to the already supported classification and search. Potential services which could be implemented are influenced by the remaining *Could Have* requirements which were not met due to time constraints. The first is *timeline generation*, which could allow a timeline to be automatically created based on critical Tweets or significant event changes posted over the duration of a crisis event. This could prove to be beneficial for response agencies to gain insight into key moments of a disaster, which could allow response tactics to be analysed, potentially aiding the response efforts of future disasters. Furthermore, an *event summarisation* service could provide similar functionalities and allow event reports to be generated after its completion. These reports could be useful for head-office response officers to gain an overview of an event, without requiring to spend time navigating the different information feeds on the system.

In the short term, Event Tracker can be made available to the participants of the TREC-IS track. Again, due to the flexible and modular architecture, microservices such as those proposed by different groups participating to the TREC-IS track could be integrated and deployed, enabling different demonstrations of said services. This could be used to analyse and evaluate the corresponding outcomes and effectiveness of different technologies, with the goal of improving the technologies and hence successfully aid the emergency management sector.

8.3 Final Reflection

This project has taught me a great deal in terms of individually organising and carrying out a large-scale piece of work. It has stressed the significance of project management and identified key processes and techniques to aid this, along with highlighting the importance of working within a set scope and time constraint. A second key takeaway from the completion of this project is the relevance of extensive user evaluations. Prior to the evaluation of Event Tracker, I was unaware of how beneficial these evaluations could be, and the usefulness of experimentation to elicit results regarding software and its effectiveness. Lastly, writing the demonstration paper regarding Event Tracker submitted to SIGIR 2019 taught me valuable lessons in academic writing and how to successfully motivate the problem in which a system is designed to solve.

A Database Schema

This appendix illustrates the database schema created for the Event Tracker relational database.

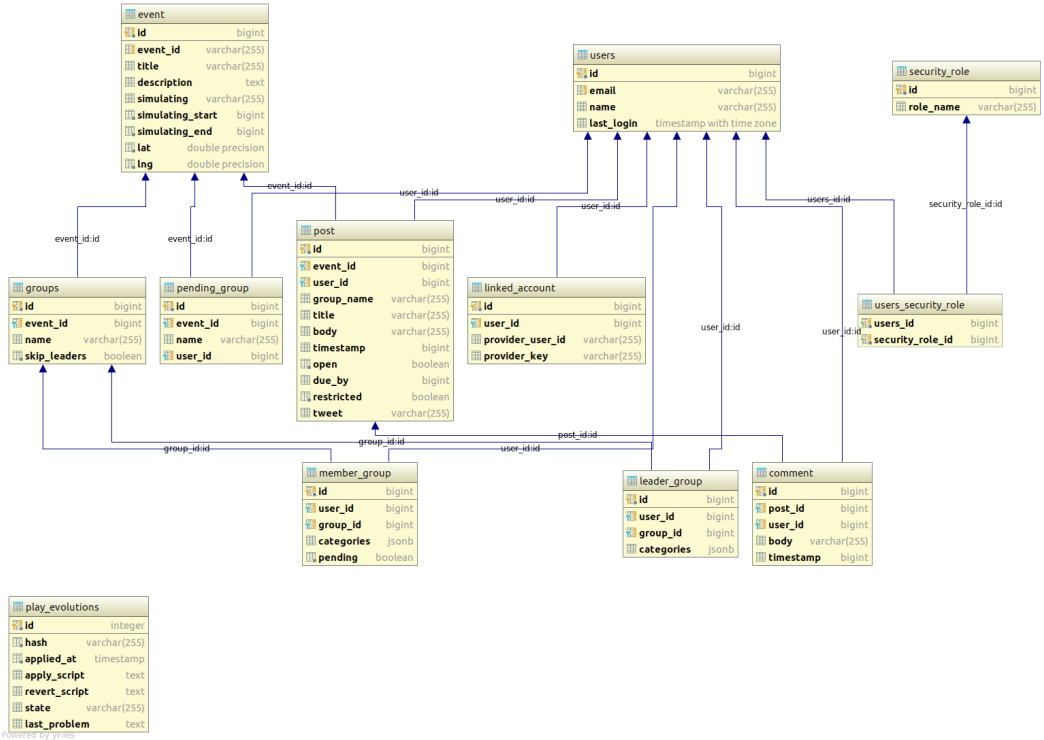


Figure A.1: *Auto-generated diagram of database schema describing the relationships between data stored in the relational database.*

B | Final Deliverable

This appendix pictures several pages of the Event Tracker platform, including the *global* and *leader* dashboard, the simulator controls and the sign up page.

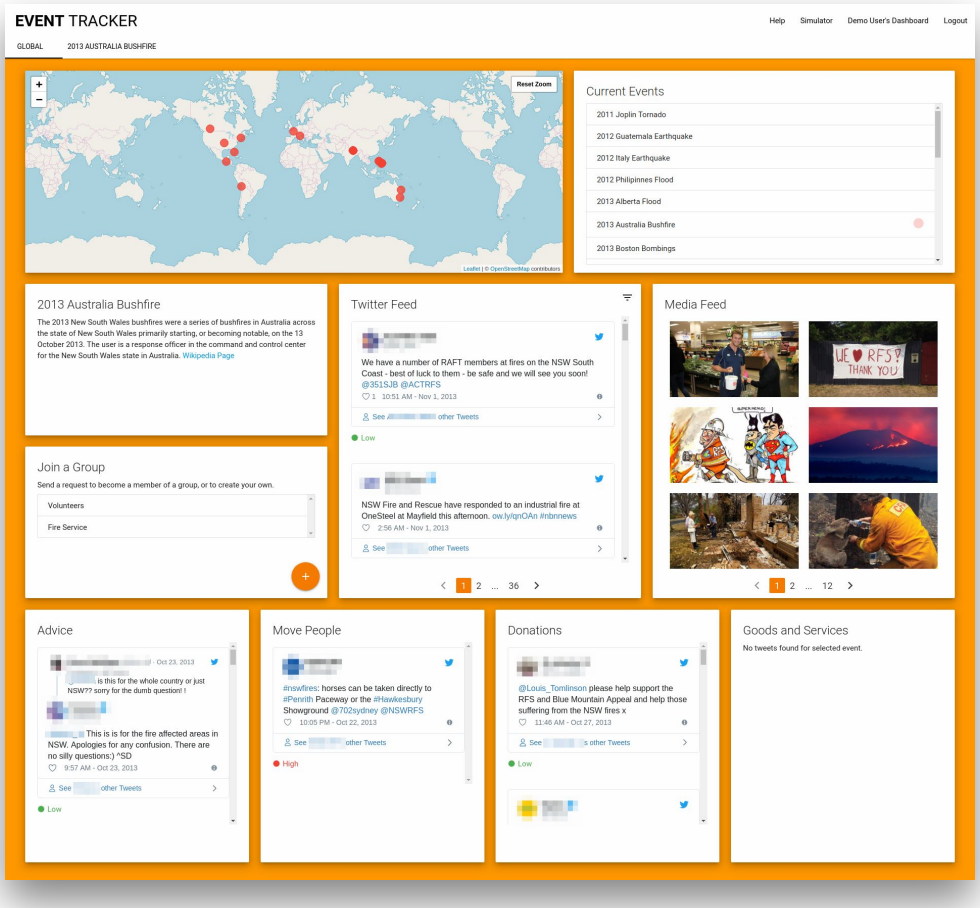


Figure B.1: *Example Global dashboard, where a user can easily switch between different events, request to join a group relating to the active event and view a subset of event details.*

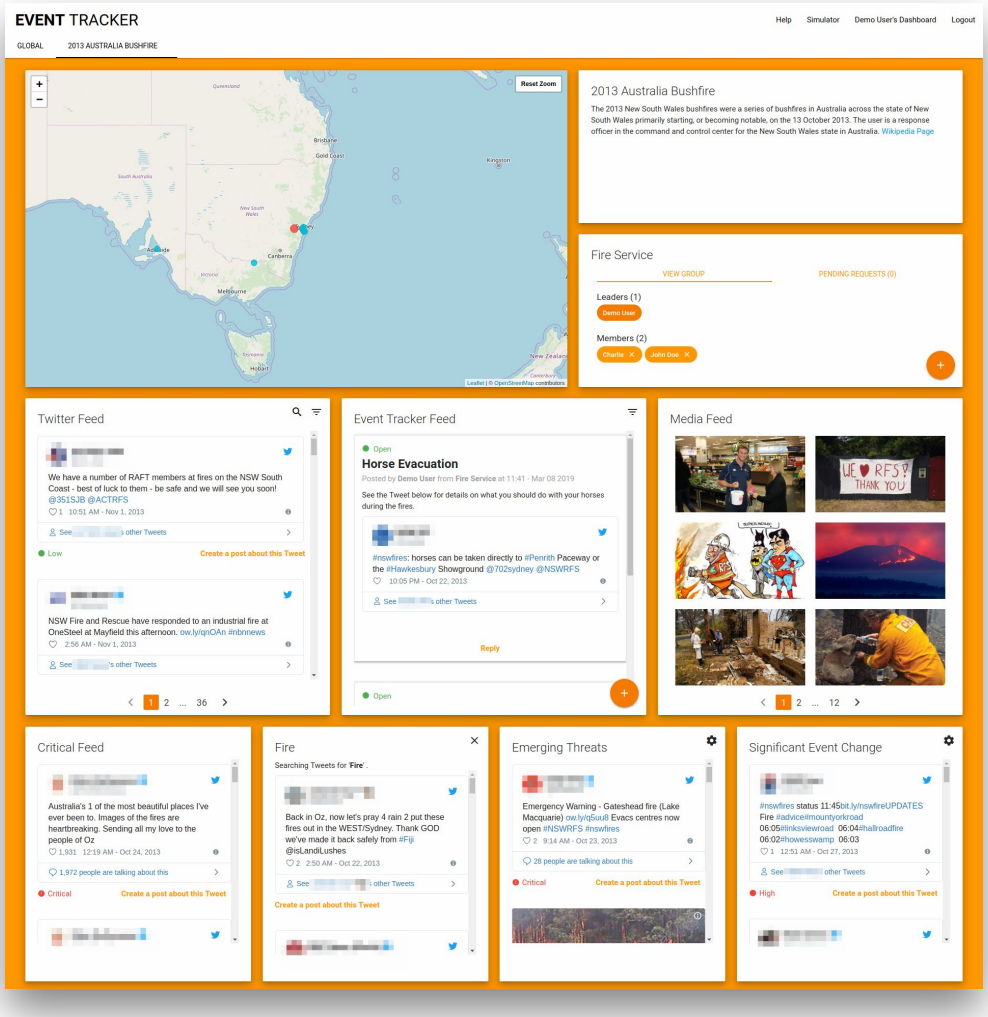


Figure B.2: *Example user dashboard, where the leader of the Fire Service group is tracking the 2013 Australia Bushfire event.*

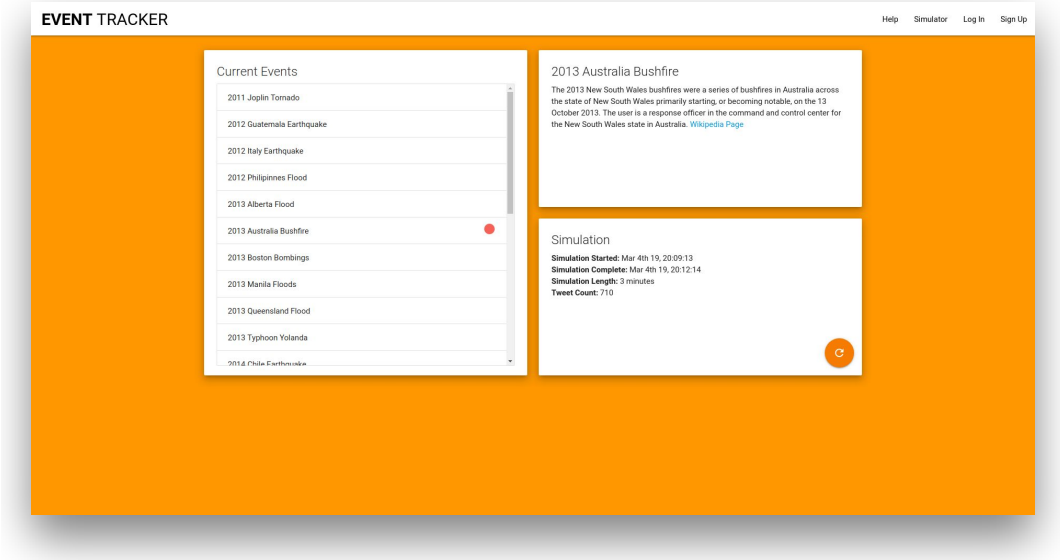


Figure B.3: *Simulator page, where users can switch between the events currently being tracked on the system and view or restart the simulation process.*

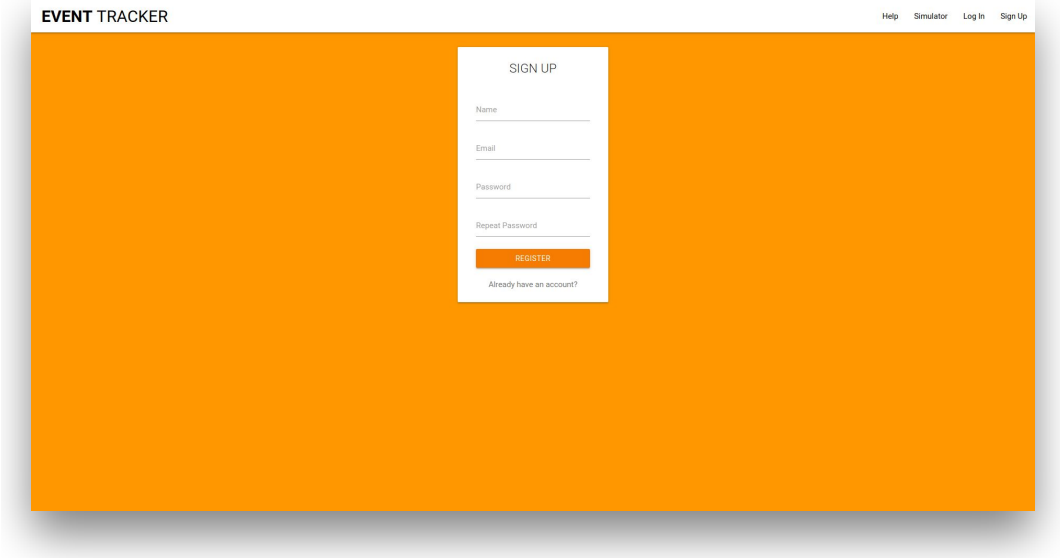


Figure B.4: *Sign up page, where users can register to join Event Tracker by filling in details such as name, email and password.*

C | Ethics Checklist

This appendix provides the signed user evaluation ethics checklist, required before user evaluation took place.

School of Computing Science
University of Glasgow

Ethics checklist for 3rd year, 4th year, MSci, MRes, and taught MSc projects

This form is only applicable for projects that use other people ('participants') for the collection of information, typically in getting comments about a system or a system design, getting information about how a system could be used, or evaluating a working system.

If no other people have been involved in the collection of information, then you do not need to complete this form.

If your evaluation does not comply with any one or more of the points below, please submit an ethics approval form to the Department Ethics Committee.

If your evaluation does comply with all the points below, please sign this form and submit it with your project.

---

1. Participants were not exposed to any risks greater than those encountered in their normal working life.
   Investigators have a responsibility to protect participants from physical and mental harm during the investigation. The risk of harm must be no greater than in ordinary life. Areas of potential risk that require ethical approval include, but are not limited to, investigations that occur outside usual laboratory areas, or that require participant mobility (e.g. walking, running, use of public transport), unusual or repetitive activity or movement, that use sensory deprivation (e.g. ear plugs or blindfolds), bright or flashing lights, loud or disorienting noises, smell, taste, vibration, or force feedback

2. The experimental materials were paper-based, or comprised software running on standard hardware.
   Participants should not be exposed to any risks associated with the use of non-standard equipment: anything other than pen-and-paper, standard PCs, mobile phones, and PDAs is considered non-standard.

3. All participants explicitly stated that they agreed to take part, and that their data could be used in the project.
   If the results of the evaluation are likely to be used beyond the term of the project (for example, the software is to be deployed, or the data is to be published), then signed consent is necessary. A separate consent form should be signed by each participant.

   Otherwise, verbal consent is sufficient, and should be explicitly requested in the introductory script.

4. No incentives were offered to the participants.
   The payment of participants must not be used to induce them to risk harm beyond that which they risk without payment in their normal lifestyle.

Figure C.1: *Signed ethics checklist, part 1 of 2*

5. No information about the evaluation or materials was intentionally withheld from the participants.
   Withholding information or misleading participants is unacceptable if participants are likely to object or show unease when debriefed.

6. No participant was under the age of 16.
   Parental consent is required for participants under the age of 16.

7. No participant has an impairment that may limit their understanding or communication.
   Additional consent is required for participants with impairments.

8. Neither I nor my supervisor is in a position of authority or influence over any of the participants.
   A position of authority or influence over any participant must not be allowed to pressurise participants to take part in, or remain in, any experiment.

9. All participants were informed that they could withdraw at any time.
   All participants have the right to withdraw at any time during the investigation. They should be told this in the introductory script.

10. All participants have been informed of my contact details.
    All participants must be able to contact the investigator after the investigation. They should be given the details of both student and module co-ordinator or supervisor as part of the debriefing.

11. The evaluation was discussed with all the participants at the end of the session, and all participants had the opportunity to ask questions.
    The student must provide the participants with sufficient information in the debriefing to enable them to understand the nature of the investigation.

12. All the data collected from the participants is stored in an anonymous form.
    All participant data (hard-copy and soft-copy) should be stored securely, and in anonymous form.

Project title Event Tracker

Student's Name Charles Thomas

Student's Registration Number 2198970T

Student's Signature Charles Thomas

Supervisor's Signature Richard McCreesh

Date 18/02/19

Figure C.2: *Signed ethics checklist, part 2 of 2*

D User Evaluation Tasks

This appendix contains the tasks which were given to participants prior to the user evaluation. Two experiments took place, each with two tasks.

D.1 Experiment 1

D.1.1 Task 1

1. You are acting as a volunteer during the 2013 Boston Bombings event.
2. Start a simulation for this event at 4x speed.
3. Create an account and log in.
4. Join the volunteer group for the 2013 Boston Bombings event.
5. Navigate to your new volunteer dashboard for the event
6. Use the modules available to find a critical priority Tweet
7. Create a post with a due date embedding this Tweet.
8. Log out of the system.

D.1.2 Task 2

1. Create a new account and log in.
2. Request to create a new group for any event.
3. Log out, and log in using the credentials:
   email: admin@test.com
   password: password
4. Accept the new group request for the event.
5. Start the search indexing for the event.
6. Log out and log back in to your previous account.
7. Visit your new dashboard.
8. Add the user 'John Doe' as a member of your group.
9. Change the visible category modules on the dashboard to 'Continuing News', 'Donations' and 'Significant Event Changes'.
10. Run a search and create a post about a Tweet you find.

D.2 Experiment 2

D.2.1 Task 1

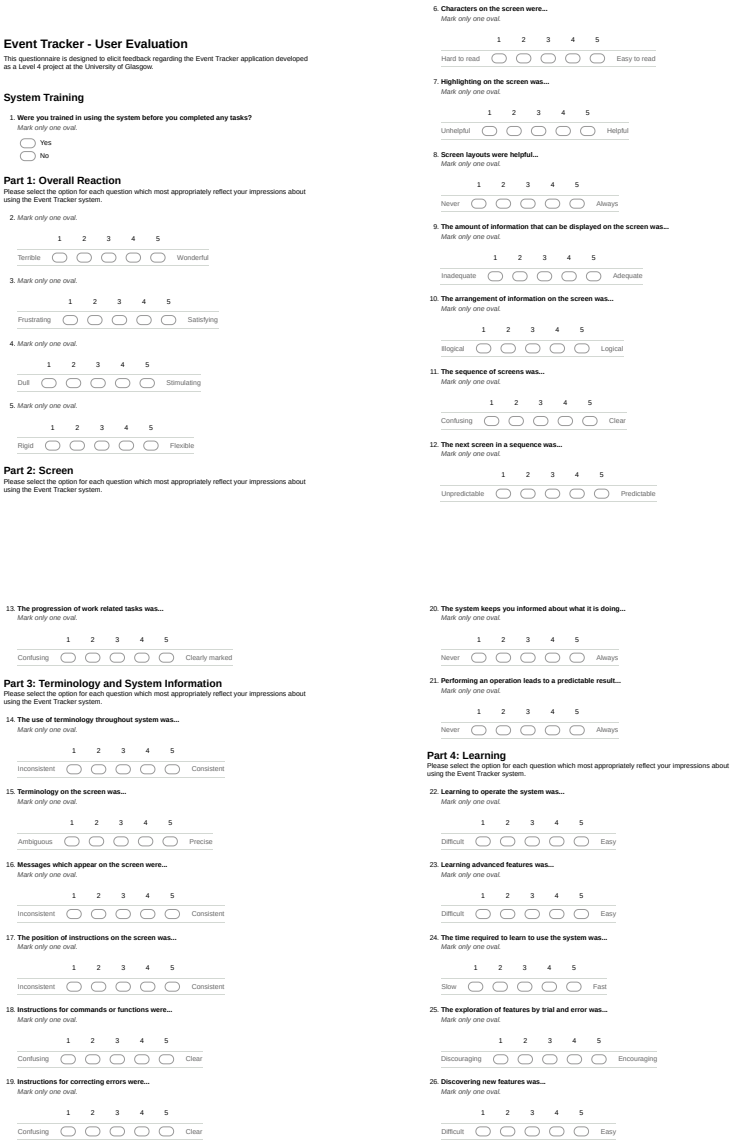
You are going to act as a volunteer highlighting critical Tweets to emergency response agencies. I will set you up as a volunteer on the 2018 Florida School Shooting event. You will then get 4 minutes to create a post about as many different Tweets that are marked **critical** as you can find. You have to specify a title and body on the posts but don't worry about what they say.

D.2.2 Task 2

You are going to act as a volunteer highlighting Tweets to emergency response agencies. I will set you up as a volunteer on the 2013 Australia Bushfire event. You will then get 5 minutes to create a post about as many different Tweets that you believe are an **emerging threat**. You have to specify a title and body on the posts but don't worry about what they say.

E | User Evaluation Questionnaire

This appendix contains the post-evaluation questionnaire completed by participants following the user evaluation experiments.



Event Tracker - User Evaluation

This questionnaire is designed to elicit feedback regarding the Event Tracker application developed as a Level 4 project at the University of Glasgow.

System Training

1. **Were you trained in using the system before you completed any tasks?**
Mark only one oval.
Yes
No

Part 1: Overall Reaction

Please select the option for each question which most appropriately reflect your impressions about using the Event Tracker system.

2. *Mark only one oval.*
1 2 3 4 5
Terrible ... Wonderful

3. *Mark only one oval.*
1 2 3 4 5
Frustrating ... Satisfying

4. *Mark only one oval.*
1 2 3 4 5
Dull ... Stimulating

5. *Mark only one oval.*
1 2 3 4 5
Rigid ... Flexible

Part 2: Screen

Please select the option for each question which most appropriately reflect your impressions about using the Event Tracker system.

6. **Characters on the screen were...**
Mark only one oval.
1 2 3 4 5
Hard to read ... Easy to read

7. **Highlighting on the screen was...**
Mark only one oval.
1 2 3 4 5
Unhelpful ... Helpful

8. **Screen layouts were helpful...**
Mark only one oval.
1 2 3 4 5
Never ... Always

9. **The amount of information that can be displayed on the screen was...**
Mark only one oval.
1 2 3 4 5
Inadequate ... Adequate

10. **The arrangement of information on the screen was...**
Mark only one oval.
1 2 3 4 5
Illogical ... Logical

11. **The sequence of screens was...**
Mark only one oval.
1 2 3 4 5
Confusing ... Clear

12. **The next screen in a sequence was...**
Mark only one oval.
1 2 3 4 5
Unpredictable ... Predictable

13. **The progression of work related tasks was...**
Mark only one oval.
1 2 3 4 5
Confusing ... Clearly marked

Part 3: Terminology and System Information

Please select the option for each question which most appropriately reflect your impressions about using the Event Tracker system.

14. **The use of terminology throughout system was...**
Mark only one oval.
1 2 3 4 5
Inconsistent ... Consistent

15. **Terminology on the screen was...**
Mark only one oval.
1 2 3 4 5
Ambiguous ... Precise

16. **Messages which appear on the screen were...**
Mark only one oval.
1 2 3 4 5
Inconsistent ... Consistent

17. **The position of instructions on the screen was...**
Mark only one oval.
1 2 3 4 5
Inconsistent ... Consistent

18. **Instructions for commands or functions were...**
Mark only one oval.
1 2 3 4 5
Confusing ... Clear

19. **Instructions for correcting errors were...**
Mark only one oval.
1 2 3 4 5
Confusing ... Clear

20. **The system keeps you informed about what it is doing...**
Mark only one oval.
1 2 3 4 5
Never ... Always

21. **Performing an operation leads to a predictable result...**
Mark only one oval.
1 2 3 4 5
Never ... Always

Part 4: Learning

Please select the option for each question which most appropriately reflect your impressions about using the Event Tracker system.

22. **Learning to operate the system was...**
Mark only one oval.
1 2 3 4 5
Difficult ... Easy

23. **Learning advanced features was...**
Mark only one oval.
1 2 3 4 5
Difficult ... Easy

24. **The time required to learn to use the system was...**
Mark only one oval.
1 2 3 4 5
Slow ... Fast

25. **The exploration of features by trial and error was...**
Mark only one oval.
1 2 3 4 5
Discouraging ... Encouraging

26. **Discovering new features was...**
Mark only one oval.
1 2 3 4 5
Difficult ... Easy

Figure E.1: *User evaluation questionnaire, part 1 of 2*

27. **Tasks can be performed in a straight-forward manner...**
Mark only one oval.

	1	2	3	4	5	
Never	◯	◯	◯	◯	◯	Always

28. **The number of steps per tasks was...**
Mark only one oval.

	1	2	3	4	5	
Too many	◯	◯	◯	◯	◯	Just right

29. **The steps to complete a task follow a logical sequence...**
Mark only one oval.

	1	2	3	4	5	
Never	◯	◯	◯	◯	◯	Always

30. **Feedback on the completion of sequence of steps was...**
Mark only one oval.

	1	2	3	4	5	
Unclear	◯	◯	◯	◯	◯	Clear

Part 5: System Capabilities

Please select the option for each question which most appropriately reflect your impressions about using the Event Tracker system.

31. **The system speed was...**
Mark only one oval.

	1	2	3	4	5	
Too slow	◯	◯	◯	◯	◯	Fast enough

32. **Response times for most operations were...**
Mark only one oval.

	1	2	3	4	5	
Too slow	◯	◯	◯	◯	◯	Fast enough

33. **The rate at which information is displayed was...**
Mark only one oval.

	1	2	3	4	5	
Too slow	◯	◯	◯	◯	◯	Fast enough

34. **The ease of operation depends on your level of experience...**
Mark only one oval.

	1	2	3	4	5	
Never	◯	◯	◯	◯	◯	Always

Part 6: Help

Please select the option for each question which most appropriately reflect your impressions about using the Event Tracker system.

35. **The amount of help given was...**
Mark only one oval.

	1	2	3	4	5	
Inadequate	◯	◯	◯	◯	◯	Adequate

36. **The placement of help messages on the screen was...**
Mark only one oval.

	1	2	3	4	5	
Confusing	◯	◯	◯	◯	◯	Clear

37. **Accessing help messages was...**
Mark only one oval.

	1	2	3	4	5	
Difficult	◯	◯	◯	◯	◯	Easy

38. **The content of the help messages was...**
Mark only one oval.

	1	2	3	4	5	
Confusing	◯	◯	◯	◯	◯	Clear

39. **The amount of help given was...**
Mark only one oval.

	1	2	3	4	5	
Inadequate	◯	◯	◯	◯	◯	Adequate

40. **Finding specific information using the help provided was...**
Mark only one oval.

	1	2	3	4	5	
Difficult	◯	◯	◯	◯	◯	Easy

Part 7: Qualitative Feedback

41. **Did you encounter any errors whilst using the Event Tracker system? If so, please specify what happened.**

42. **Do you have any suggestions on how the system could be improved?**

43. **Do you think that Event Tracker could be a useful tool for emergency management agencies to monitor events? If no, please specify why not.**

44. **Do you think that Event Tracker could be a useful tool for volunteers to help during a disaster? If no, please specify why not.**

45. **Do you think that Event Tracker could be a useful tool in providing victims during a disaster advice and guidance on steps they should take? If no, please specify why not.**

46. **Do you have any other comments about the system?**

Figure E.2: *User evaluation questionnaire, part 2 of 2*

F Requirement Validation Evidence

F.1 Functional Requirements

Requirement	Priority	Met?	Reasoning
1	MH	✓	Met by user authentication
2	MH	✓	Met by public and global dashboards
3	MH	✓	Met by the Current Event feed and the Event Map modules
4	MH	✓	Met by admin dashboard
5	MH	✓	Met by admin group management
6	MH	✓	Met by admin group management
7	SH	✓	Met by admin group management
8	MH	✓	Met by leader dashboard
9	MH	✓	Met by leader group management
10	MH	✓	Met by leader and member dashboards
11	MH	✓	Met by the Event Tracker Feed module
12	SH	✓	Met by 'Create a Group' functionality
13	SH	✓	Met by a public option when creating a post on the Event Tracker Feed
14	SH	✓	Met by the Tweet Category feeds and integration with classification service
15	SH	✓	Met by the Twitter Feed filtering, Critical Feed and integration with classification service
16	SH	✓	Met by the Twitter Feed module and integration with the search service
17	SH	✓	Met by the Critical Feed module
18	CH	✓	Met by Single Event Map module
19	CH	×	Not met
20	CH	×	Not met
21	CH	×	Not met
22	WH	×	Outwith project scope
23	MH	✓	Met by 'Join a Group' functionality and bypassing request stage for automatically created volunteers group
24	MH	✓	Met by the Event Tracker Feed module
25	SH	✓	Met by the Tweet Category modules
26	MH	✓	Met by the public dashboard
27	MH	✓	Met by the Tweet Category modules and the public option on the Event Tracker Feed module

Table F.1: *Functional requirement validation evidence*

F.2 Non-Functional Requirements

Requirement	Priority	Met?	Reasoning
28	MH	✓	Met as system was designed using a flexible architecture
29	MH	✓	Met as application is fully response and no functionality is disabled on small devices
30	MH	✓	Met by using modern frameworks which work on all modern browsers
31	MH	✓	Met as proven in user evaluation
32	MH	✓	Met as proven in performance and user evaluations
33	MH	✓	Met by building system using Play Framework

Table F.2: *Non-functional requirement validation evidence*

G Demonstration Paper

The following pages contain the demonstration paper titled *Event Tracker: A Text Analytics Platform for Use During Disasters* submitted to SIGIR 2019.

Event Tracker: A Text Analytics Platform for Use During Disasters

Charles Thomas
2198970t@student.gla.ac.uk
University of Glasgow, UK

Richard McCreadie
richard.mccreadie@glasgow.ac.uk
University of Glasgow, UK

Iadh Ounis
iadh.ounis@glasgow.ac.uk
University of Glasgow, UK

ABSTRACT

Emergency management organisations currently rely on a wide range of disparate tools and technologies to support the monitoring and management of events during crisis situations. This has a number of disadvantages, in terms of training time for new staff members, reliance on external services, and a lack of integration (and hence poor transfer of information) between those services. On the other hand, Event Tracker is a new solution that aims to provide a unified view of an event, integrating information from emergency response officers, the public (via social media) and also volunteers from around the world. In particular, Event Tracker provides a series of novel functionalities to realise this unified view of the event, namely: real-time identification of critical information, automatic grouping of content by the information needs of response officers, as well as real-time volunteers management and communication. This is supported by an efficient and scalable back-end infrastructure designed to ingest and process high-volumes of real-time streaming data with low latency.

ACM Reference Format:
Charles Thomas, Richard McCreadie, and Iadh Ounis. 2019. Event Tracker: A Text Analytics Platform for Use During Disasters. In *Proceedings of SIGIR '19: 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

A 68% rise in the use of social networking sites since 2005 [15] has introduced an abundance of real-time information online. This has enabled new ways for the public to contact response agencies [4], and grants those in the Emergency Management sector a new means of accessing potentially life-saving information. With a three-fold increase in natural disasters over the past 35 years [11], it is extremely important that emergency response agencies have the tools available to ensure that they can monitor social media streams in real-time and provide assistance to the public in a quick and effective manner.

Data posted on social media platforms regarding a disaster or world events could potentially provide a wide range of valuable information to emergency response services. For example, responding to cries for help, accessing first-hand observations or simply gaining an insight into public opinions could provide an added-value to these agencies. However, with the wealth of information now available, it is critical that the data can be easily visualised and navigated, ensuring that emergency services can quickly find and act upon key information.

There also lies potential in the connection of volunteers with formal emergency response officers during crisis situations. Systems designed to assist emergencies services during crises face typically a number of challenges, including a lack of coordination and communication between the officers and other formal entities as well as an unwillingness to engage and form relationships with digital volunteer groups [19]. A successful connection with both physical and digital volunteers could indeed aid response efforts immensely, with a wealth of potential skills and resources becoming available [20].

A range of systems have been developed in the past with the goal to support emergency response efforts during disasters [4]. These systems, however, have had little impact on the sector. It has been found that reasons behind this include the insufficient training of personnel, which require time to make adequate use of the platforms, the lack of guidelines for their use, and apprehension over social media trustworthiness [9].

In this paper, we propose a new system, Event Tracker, which aims to support the monitoring and management of events during disasters and crisis situations. The system intends to support three primary tiers of users - emergency response officers and their teams; physical and digital volunteers; as well as the victims during the crisis. Indeed, we argue that emergency response agencies should be given the tools to navigate large volumes of social media data with ease, supported by functionalities such actionable information feeds and criticality estimation. Moreover, volunteers should be able to provide information directly to these response agencies, either using their first-person knowledge of the situation, or aiding the data navigation and highlighting specific information and relevant details. Finally, victims during a disaster should be able to straightforwardly access advice and information from emergency services, ensuring that they are equipped with up-to-date information that could lead to their safety.

2 RELATED WORK

To-date, there have been a range of techniques and systems proposed to support crisis management. Below we provide a brief overview of key technologies and initiatives that are relevant to our proposed Event Tracker system.

Finding Relevant Information: A core function of any crisis management solution is getting key information to the right people.

SIGIR '19, July 21–25, 2019, Paris, France

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

This is operationalised either via volunteer efforts [6] or through automatic technologies [18] for categorising and prioritising reports. Indeed, a survey [5] of approaches identified eight dimensions of categorisation, namely: by information provided/contained; fact vs. subjective vs. emotional content; by information source; by credibility; by time; by location; by embedded links; or by environmental relevance (physical, built or social). Automatic categorisation efforts have focused on supervised learning technologies, often backed by human annotation of crisis data [12]. Building on this work, Event Tracker provides fully automatic real-time reports alerting that are predicted to contain critical and actionable information, as well as state-of-the-art content-based categorisation based on a crisis event ontology (from TREC-IS, which we discuss below).

Crisis Content Processing and TREC-IS: As discussed above, a range of automatic technologies to support crisis management have previously been proposed. However, individually they have had little impact on crisis management as a whole [16]. One reason for this lack of impact is data quality [10]. To avoid a similar fate, Event Tracker's automatic content categorisation service targets a crisis ontology being maintained by an active initiative, namely TREC Incident Streams (http://trecis.org). TREC-IS is a new track at the Text Retrieval Conference (TREC) started in 2018 and designed to unite academia and industry around research into automatically processing social media streams during emergency situations and categorising information and aid requests made on social media for emergency service operators. TREC-IS provides training datasets for training crisis content categorisation systems, as well as an on-going (bi-yearly) evaluation challenge aimed at increasing the TRL of such systems.

Related Systems: Before discussing the design and architecture of Event Tracker, it is worth highlighting some related systems that we either build upon or learn from. First, *EAIMS* [13], was a prototype crisis management system that aimed at exploiting social media data to provide real-time detection of emergency events (demoed at SIGIR 2016), along with search, sentiment analysis, discussion-thread extraction and summarisation functionalities. The main drawback of EAIMS is that it was primarily designed only for use by high-level, head-office, emergency response staff, in contrast to Event Tracker that also targets volunteer groups. *Twitter Moments* is a relatively new feature of the Twitter platform, added in 2015. A 'Moment' is a user curated collection of Tweets, which allows users to comment on stories, promote news or create round-ups of different world events [2]. Twitter Moments has great potential for providing a high-level overview of an event. However, a key downfall is that Moments are popularity-driven, so local content such as individual calls for help will be missed. Instead, Event Tracker relies on recall-focused crawling in conjunction to automatic content categorisation to find actionable information quickly and accurately. Furthermore, *AIDR* is a platform developed at the Qatar Computing Research Institute that collects and classifies crisis-related Tweets [12]. AIDR makes use of crowdsourcing to label Tweets, which can then be used to train an automatic classifier. This ensures that the system performs well for each new event registered, since automatic classification using only pre-trained models can lead to low accuracy. Currently, Event Tracker relies on pre-trained classification models, however its underlying microservice architecture is flexible and allows for the integration of additional services like AIDR. Finally, *Twitcident* is another web-based system for filtering and analysing social media data on real-world incidents, such as disasters. Similar to EAIMS, Twitcident [1] provides incident detection and search technologies, enabling emergency services to filter data to aid their response efforts. However, a relatively recent study showed that Twitcident did not provide much value to response agencies in practice, as it could not provide early warnings regarding critical information [3]. For this reason, Event Tracker integrates criticality estimation tools to provide real-time reporting of crucial information.

3 EVENT TRACKER ARCHITECTURE

As argued earlier, the goal of Event Tracker is to provide an integrated platform that provides both automated low-latency ingestion and augmentation of report streams (either manually entered or crawled from social media) with effective support tools that enable response officers and volunteers to collaborate together to generate a unified operational picture during an emergency. In effect, this means that the platform must support: 1) low-level support for different report streams (news reports, social media and manual form-filling by call-centre operators), 2) integration of fast text processing technologies for real-time report tagging, as well as both 3) classical on-request information access (search and content look-up) and 4) continuous push notification servicing.

As such, Event Tracker uses a flexible architecture as illustrated in Figure 2. The lowest ingestion layer provides multi-stream integration, using a common data back-end to [13]. Above this sits the augmentation layer, which enables modular and scalable integration of report tagging microservices (in this case for identification of actionable information and information criticality analysis) using a distributed Apache Flink back-end. The output of the augmentation layer is then processed by the activity layer, that handles the 'business logic' for the application (data and user management), feeding either persistent storage structures (a search index and database) and/or the front-end directly via push notifications. Indeed, during a disaster event, the volume and rate at which relevant content that needs to be processed may vary greatly (anywhere from 10 posts/min to 4,000 posts/min [7]). Thus, the architecture behind the Event Tracker is designed to handle high-volume streams of data.

4 KEY FUNCTIONALITIES

Event Tracker integrates a number of functionalities following the design vision discussed earlier. Below, we summarise these main functionalities.

Generating Actionable Information Feeds: As discussed above, getting key information to the right people is crucial for a successful crisis management system, and one of the main reasons why previous systems have failed. Event Tracker combats this by providing multiple actionable information feeds through user dashboards, each of which is designed with a different intention of the value it can provide to the response agencies. Figure 1 (a) pictures the *Twitter Feed*, a real-time, filterable collection of all data gathered across the duration of a given crisis event. The *Media Feed*, displayed in Figure 1 (d), extracts all media collected, enabling response officers

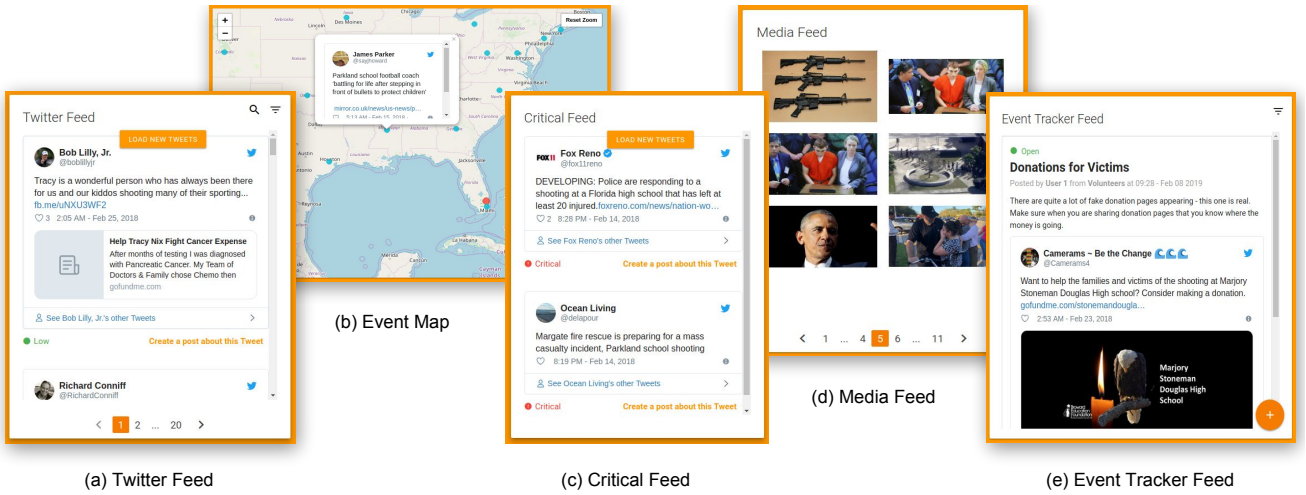


Figure 1: A selection of different functionalities available on the dashboards on Event Tracker

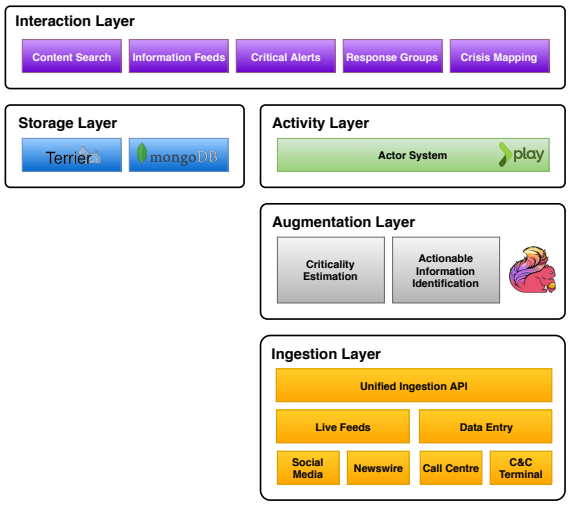


Figure 2: Conceptual architecture

to, for example, assess damage during and after a disaster. Automatic content-based categorisation, through the TREC-IS initiative, examined in Section 2, is exploited by Event Tracker to provide emergency response agencies with a set of *Tweet Category Feeds*, in an attempt to increase how quickly and accurately these agencies can navigate the collection of data to find actionable information. Different agencies with different motivations can customise the set of feeds to distinctive categories such as 'First Party Observation' or 'Official Reports'. Each feed on Event Tracker aims to improve the situational awareness of the agencies, which is vital in effective decision making in complex and dynamic environments [8].

Criticality Alerting: Early warnings and alerts regarding critical information posted online during a crisis situation is imperative for a quick response from emergency services. Aid requests could be answered quickly, increasing for example the likelihood that a victim could be assisted in time. Event Tracker aims to accomplish this by also making use of the TREC-IS initiative to provide real-time identification of critical information. Fast classification and text processing technologies are exploited to immediately label incoming messages with a criticality score of *low*, *medium*, *high*, or *critical*, which can then be displayed alongside Tweets on a user's dashboard. Continuous push notifications servicing is also harnessed by Event Tracker to populate the *Critical Feed*, pictured in Figure 1 (c), to highlight reports which are predicted to contain critical and immediately actionable information.

Communicating Amongst Response Associations & Groups: To support crisis management, there are a large number of associations that could help monitor and manage a crisis (from the Red Cross to smaller Regional Volunteer Groups). Each event being tracked on the system can be related to multiple response associations, each of which may provide support in different manners. These associations can communicate through Event Tracker, using the *Event Tracker Feed* (Figure 1 (e)). Members of the response teams can use this feed to communicate directly with other response agencies, passing on any relevant information. As discussed, there has been a lack of successful systems that integrate formal response agencies with volunteers, and so for each event registered on Event Tracker, an open volunteering group is created, which anyone can join. The system can be used to increase the coordination and communication between these entities, with both volunteers on the ground reporting information, or digital volunteers highlighting critical reports to official response officers by embedding Tweets into their posts.

Additional Functionalities: Along with these key functionalities, Event Tracker provides some other features which are worth mentioning. First, we describe the *Crisis mapping* functionality.Figure 1 (b) pictures the *Event Map* module on Event Tracker, which displays both the location of the related event and the geo-tagged Tweets that have been collected over the duration of the crisis. As the information feeds, this aids in increasing the situational awareness of the response agencies. The module could be extended in the future to

display the locations of other reports, to improve data management. Next, we describe the *On-request information access* functionality. Making use of the Terrier search engine [14], Event Tracker allows users to explore the collected data with ease. New feeds are created on the user's dashboard, enabling multiple searches to run concurrently. For example, emergency response agencies can make use of this feature to discover reports relating to specific queries, such as 'fire'.

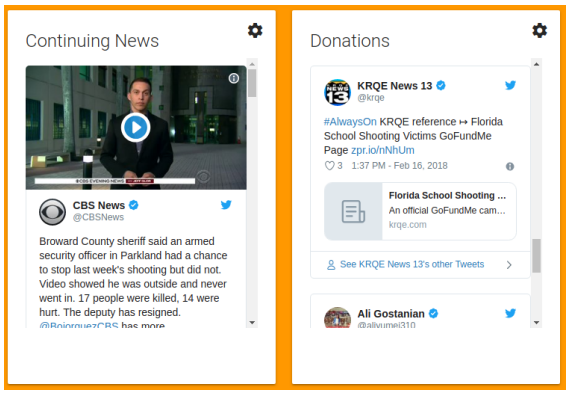


Figure 3: Potential Tweet Category feeds on user dashboards

5 ILLUSTRATIVE USE CASES

Focussing on multiple types of users during a crisis event, Event Tracker has several use cases. To illustrate the working of Event Tracker, let's consider two possible user cases. In the first use case, consider an emergency response officer, who uses the system to view all the actionable information feeds and navigate the incoming reports of data. During a simulation of the 2013 Australia Bushfire crisis, Event Tracker marked the Tweet "*Horrid emergency unfolding at #faulconbridge - fire is 50m behind #springwood shs - near hundreds of homes, norman lindsay gallery #nswfires*" as a high criticality emerging threat. Had Event Tracker been deployed over this event, a response officer accessing this report could have contributed an effective response. During an event, these officers may be monitoring multiple communication channels [17], which may lead to sporadic focus on each of the information feeds. As a consequence, some important information about tnhe event might be missed. Instead, with Event Tracker, the criticality alerting component would notify the user with any new reports that are predicted to be of vital importance, ensuring that the end-user can act quickly and efficiently. As a second case, consider a volunteer on the ground during a disaster. They could make use of Event Tracker to coordinate with response agencies to ensure that they are providing the best help they can. They can also make reports for the response agencies, to bring any significant information they possess to the attention of the response officers.

6 CONCLUSIONS

In this paper, we have presented Event Tracker, a modular and extensible prototype system designed to support the monitoring and management of events during crisis situations. The system leverages a flexible architecture, which is designed for low latency, and high-volume streams of data to provide functionalities such as actionable information feeds and criticality alerting. Event Tracker also enables the communication between various emergency response agencies and volunteers, and provides additional functionalities such as crisis mapping and on-request information access. In the short term, we aim to make Event Tracker available to the participants of the TREC-IS track. In particular, due to the modular augmentation layer Event Tracker is built upon, microservices such as those proposed by different groups participating to the TREC-IS track could be integrated and deployed, enabling different demonstrations to the end-users, whereby the corresponding outcomes and effectiveness of different technologies can be evaluated and further investigated.

REFERENCES

[1] Fabian Abel, Claudia Hauff, Geert-Jan Houben, Richard Stronkman, and Ke Tao. 2012. Twitcident: Fighting fire with information from Social Web streams. *WWW'12 - Proceedings of the 21st Annual Conference on World Wide Web Companion* (04 2012). https://doi.org/10.1145/2187980.2188035
[2] Liz Alton. 2018. Everything you need to know about Twitter Moments. https://business.twitter.com/en/blog/Everything-you-need-to-know-about-Twitter-Moments.html. Accessed: 09.02.2019.
[3] Kees Boersma, Dominique Diks, Julie Ferguson, and Jeroen Wolbers. 2016. *From Reactive to Proactive Use of Social Media in Emergency Response: A Critical Discussion of the Twitcident Project.* https://doi.org/10.4018/978-1-4666-9867-3.ch014
[4] Carlos Castillo. 2016. *Big Crisis Data: Social Media in Disasters and Time-Critical Situations.* Cambridge University Press. https://doi.org/10.1017/CBO9781316476840
[5] Carlos Castillo. 2016. *Big crisis data: social media in disasters and time-critical situations.* Cambridge University Press.
[6] Lise Ann St Denis, Amanda L Hughes, and Leysia Palen. 2012. Trial by fire: The deployment of trusted digital volunteers in the 2011 shadow lake fire. In *Proceedings of ISCRAM.*
[7] R. McCreadie et al. 2016. D4.7 - Integrated Search over Social Media. *Deliverable, SUPER FP7 Project* (2016).
[8] J. Harrald and T. Jefferson. 2007. Shared Situational Awareness in Emergency Management Mitigation and Response. In *2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07).* 23–23. https://doi.org/10.1109/HICSS.2007.481
[9] Starr Hiltz, Jane Kushma, and Linda Plotnick. 2014. Use of Social Media by U.S. Public Sector Emergency Managers: Barriers and Wish Lists. https://doi.org/10.13140/2.1.3122.4005
[10] Starr Roxanne Hiltz, Jane A Kushma, and Linda Plotnick. 2014. Use of Social Media by US Public Sector Emergency Managers: Barriers and Wish Lists.. In *Proceedings of ISCRAM.*
[11] Peter Hoeppe. 2015. Trends in weather related disasters - Consequences for insurers and society. (2015).
[12] Muhammad Imran, Carlos Castillo, Ji Lucas, Patrick Meier, and Sarah Vieweg. [n. d.]. AIDR: Artificial intelligence for disaster response. In *Proceedings of WWW.*
[13] Richard McCreadie, Craig Macdonald, and Iadh Ounis. 2016. EAIMS: Emergency Analysis Identification and Management System. https://doi.org/10.1145/2911451.2911460
[14] Iadh Ounis, Gianni Amati, Vassilis Plachouras, Ben He, Craig MacDonald, and Christina Lioma. 2006. Terrier : A High Performance and Scalable Information Retrieval Platform. In *Proceedings of OSIR'2006.*
[15] Andrew Perrin. 2015. Social Media Usage: 2005-2015. Accessed: 16.02.2019.
[16] C Reuter, G Backfried, MA Kaufhold, and F Spahr. 2018. ISCRAM turns 15: A Trend Analysis of Social Media Papers 2004-2017. In *Proceedings of ISCRAM.*
[17] Andrea H Tapia, Kathleen A Moore, and Nichloas J Johnson. 2013. Beyond the trustworthy tweet: A deeper understanding of microblogged data use by disaster response and humanitarian relief organizations.. In *Proceedings of ISCRAM.*
[18] Marie Truelove, Maria Vasardani, and Stephan Winter. [n. d.]. Towards credibility of micro-blogs: characterising witness accounts. *GeoJournal* ([n. d.]).
[19] J Twigg and Irina Mosel. 2017. Emergent groups and spontaneous volunteers in urban disaster response. *Environment and Urbanization* 29 (08 2017), 095624781772141. https://doi.org/10.1177/0956247817721413
[20] Joshua Whittaker, Blythe McLennan, and John Handmer. 2015. A review of informal volunteerism in emergencies and disasters: Definition, opportunities and challenges. *International Journal of Disaster Risk Reduction* 13 (2015), 358 – 368. https://doi.org/10.1016/j.ijdrr.2015.07.010

Bibliography

F. Abel, C. Hauff, G.-J. Houben, R. Stronkman, and K. Tao. Twitcident: Fighting fire with information from social web streams. *WWW'12 - Proceedings of the 21st Annual Conference on World Wide Web Companion*, 04 2012. doi: 10.1145/2187980.2188035.

L. Alton. Everything you need to know about twitter moments. `https://business.twitter.com/en/blog/Everything-you-need-to-know-about-Twitter-Moments.html`, 2018. Accessed: 09.02.2019.

Angular. Angular. `https://angular.io/`, 2019. Accessed: 22.03.2019.

Apache. Apache solr. `http://lucene.apache.org/solr/`, 2019. Accessed: 22.03.2019.

K. Boersma, D. Diks, J. Ferguson, and J. Wolbers. *From Reactive to Proactive Use of Social Media in Emergency Response: A Critical Discussion of the Twitcident Project*. 01 2016. ISBN 978-1-4666-9867-3. doi: 10.4018/978-1-4666-9867-3.ch014.

C. Castillo. *Big Crisis Data: Social Media in Disasters and Time-Critical Situations*. Cambridge University Press, 2016. doi: 10.1017/CBO9781316476840.

A. B. Consortium. Moscow prioritisation. `https://www.agilebusiness.org/content/moscow-prioritisation`, 2014. Accessed: 16.01.2019.

Docker. Docker compose. `https://docs.docker.com/compose/`, 2018. Accessed: 22.03.2019.

Docker. Docker. `https://www.docker.com/`, 2019. Accessed: 22.03.2019.

Ebean. Ebean orm. `https://ebean.io/`, 2019. Accessed: 22.03.2019.

GitHub. Github. `https://github.com/`, 2019. Accessed: 22.03.2019.

GIVE.asia. sbt-vuefy. `https://github.com/GIVESocialMovement/sbt-vuefy`, 2019. Accessed: 23.03.2019.

Google. Lighthouse. `https://developers.google.com/web/tools/lighthouse/`, 2018. Accessed: 08.03.2019.

B. D. Harper and K. L. Norman. Improving user satisfaction: The questionnaire for user interaction satisfaction version 5.5. In *Proceedings of the 1st Annual Mid-Atlantic Human Factors Conference*, pages 224–228, 1993.

J. Harrald and T. Jefferson. Shared situational awareness in emergency management mitigation and response. In *2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07)*, pages 23–23, Jan 2007. doi: 10.1109/HICSS.2007.481.

S. Hiltz, J. Kushma, and L. Plotnick. Use of social media by u.s. public sector emergency managers: Barriers and wish lists. 01 2014. doi: 10.13140/2.1.3122.4005.

P. Hoeppe. Trends in weather related disasters - consequences for insurers and society. 2015.

M. Imran, C. Castillo, J. Lucas, P. Meier, and S. Vieweg. Aidr: Artificial intelligence for disaster response. In *Proceedings of WWW*. ACM, 2014.

JaCoCo. Jacoco java code coverage library. https://github.com/jacoco/jacoco, 2019. Accessed: 22.03.2019.

JReport. 3-tier architecture: A complete overview. https://www.jinfonet.com/resources/bi-defined/3-tier-architecture-complete-overview/, 2018. Accessed: 18.01.2019.

M. Kardara and J. Violos. D5.4 - modular integration of social sensors, social media platforms and social network. *Deliverable, SUPER FP7 Project*, 2015.

A. Koutifaris. Test driven development: what it is, and what it is not. https://medium.freecodecamp.org/test-driven-development-what-it-is-and-what-it-is-not-41fa6bca02a2, 2018. Accessed: 22.03.2019.

Leaflet. Leaflet. https://leafletjs.com/, 2019. Accessed: 22.03.2019.

Materialize. Materialize. https://materializecss.com/, 2019. Accessed: 22.03.2019.

R. McCreadie, C. Macdonald, and I. Ounis. Eaims: Emergency analysis identification and management system. 2016. doi: 10.1145/2911451.2911460.

R. McCreadie, C. Buntain, and I. Soboroff. Trec incident streams: Finding actionable information on social media. In *Proceedings of ISCRAM*, 2019.

MongoDB. Mongodb. https://www.mongodb.com/, 2019. Accessed: 22.03.2019.

A. Moralis. D5.6 - super middleware platform. *Deliverable, SUPER FP7 Project*, 2016.

Y. Navada. What are websockets? https://medium.com/front-end-weekly/what-are-websockets-7bf0e2e1af2, 2018. Accessed: 19.03.2019.

OpenStreetMap. Openstreetmap. https://www.openstreetmap.org, 2019. Accessed: 22.03.2019.

I. Ounis, G. Amati, V. Plachouras, B. He, C. MacDonald, and C. Lioma. Terrier : A high performance and scalable information retrieval platform. In *Proceedings of OSIR'2006*, 2006.

I. Ounis, C. Macdonald, and R. McCreadie. D4.7 - integrated search over social media. *Deliverable, SUPER FP7 Project*, 2016.

A. Perrin. Social media usage: 2005-2015, 2015. Accessed: 16.02.2019.

Play Framework. Play framework. https://www.playframework.com/, 2019. Accessed: 22.03.2019.

PostgreSQL. Postgresql: The world's most advanced open source relational database. https://www.postgresql.org/, 2019. Accessed: 22.03.2019.

RabbitMQ. Rabbitmq. https://www.rabbitmq.com/, 2019. Accessed: 22.03.2019.

K. Reen. 5 tips for using twitter during emergencies and natural disaster. https://blog.twitter.com/official/en_sea/topics/insights/2018/5-Tips-for-using-Twitter-during-emergencies-and-natural-disaster.html, 2018. Accessed: 09.02.2019.

S. Robertson and J. Robertson. *Mastering the Requirements Process: Getting Requirements Right*. Addison-Wesley Professional, 2012. ISBN 0321815742 9780321815743.

SIGIR. Special interest group on information retrieval. http://sigir.org/, 2019. Accessed: 23.03.2019.

SUPER. Super. http://super-fp7.eu/, 2017. Accessed: 22.03.2019.

Travis-CI. Travis ci. https://travis-ci.org/, 2019. Accessed: 22.03.2019.

Trello. Trello. https://trello.com/, 2019. Accessed: 22.03.2019.

J. Twigg and I. Mosel. Emergent groups and spontaneous volunteers in urban disaster response. *Environment and Urbanization*, 29:095624781772141, 08 2017. doi: 10.1177/0956247817721413.

Twitter. Display requirements: Tweets. https://developer.twitter.com/en/developer-terms/display-requirements.html, 2019. Accessed: 19.03.2019.

Vue.js. Vue.js. https://vuejs.org/, 2019a. Accessed: 22.03.2019.

Vue.js. What is vuex? https://vuex.vuejs.org/, 2019b. Accessed: 22.03.2019.

J. Whittaker, B. McLennan, and J. Handmer. A review of informal volunteerism in emergencies and disasters: Definition, opportunities and challenges. *International Journal of Disaster Risk Reduction*, 13:358 – 368, 2015. ISSN 2212-4209. doi: https://doi.org/10.1016/j.ijdrr.2015.07.010. URL http://www.sciencedirect.com/science/article/pii/S2212420915300388.

H. Zade, K. Shah, V. Rangarajan, P. Kshirsagar, M. Imran, and K. Starbird. From situational awareness to actionability: Towards improving the utility of social media data for crisis response. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW), 2018.