Dissertation for Online Project in Information Technology

Sarah Rudston & Man Kit Liew

19th June 2020

Word count: 19,618

Master of Science in Information Technology
University of Aberdeen

Table of Contents

List of Figures

List of Tables

Introduction

This dissertation comprises the development of a web application to gather data and display information regarding the events and activities of the activist collective Extinction Rebellion (also known as XR) (Extinction Rebellion website, 2020). The aim of this application is to provide a single-shot location for information on events representing the UK-based segment of the network.

Extinction Rebellion began as a response to an IPCC report which stated that we have only 12 years to halt climate change on a catastrophic level. They operate on a global basis, with groups organising campaign events at a city- or even neighbourhood focused level. Many of these events are advertised on Facebook where they are linked to public and private groups comprised of XR members.

Our application's inventory of the organisation's events is designed to inform members of the public and existing activists within XR of upcoming events which they can attend, including information on where the events are taking place and the nature of what is involved. This is important as the organisation has many different events and activities running simultaneously, large and small, and the dissemination of information often leads to confusion and misinformation being spread across members. A single place for information to be shared reduces the risk of duplicate resources as well as fatigue from activists who are currently required to check multiple places for information.

As well as this, the application is aimed at engaging and inviting members of the public to become more involved in XR events without feeling that they need to join a local group or Facebook group first. The application also allows administrative privileges to members of XR in order for them to contribute and amend information about events.

In order to complete the development of the application, we acted as an affinity group joined on a temporary basis with XR's tech team (discussed in more detail later in this dissertation). Our contact at XR, Carl Hughes, was able to provide information about requirements for the application and how the organisation works to produce and integrate new technical tools.

The event data for the application itself is pulled from multiple XR pages represented on Facebook using Facebook's API. We also made the event information available through an RSS feed, meaning that multiple other XR digital resources could also display live information on events if they wish. We also developed the application so that it can be used and adapted by a small group of web developers who currently volunteer for XR, so that changes can be made for their own purposes. As such, the application utilises well-documented and flexible services including Ruby on Rails, Postgres db and Google Maps integration along with the templating library bootstrap.

The main development of this application consisted of mechanisms to pull data from Facebook, display the information and allow the end users to sort and filter the event details based on what was most relevant to them. The primary goal was to implement these three facets successfully before adding additional functionality to provide a more well-rounded application and lay the groundwork for future development.

During the period of development, the COVID-19 crisis put the majority of the UK into lockdown, meaning that many of XR's events and campaigning activities were cancelled. The organisation continued to organise and advertise events and meetings which took place entirely online, meaning that the original purpose for our application was still preserved.

At the close of development, we were able to implement all of the initial requirements for the application. The application is able to display, search and filter events as well as

having the capability to refresh information directly from Facebook. Information can also be edited by those logged in to the application as administrators. The client is satisfied that development of new features can be continued by the technical team at XR.

Structure of the Dissertation Document

The structure of the dissertation is as follows:

- Introduction to the client - This section provides some context as to why the project was chosen and why the application was required.
- The project development process - This section details the rationale behind our development process.
- Requirements gathering and development of user stories - This section outlines the use cases for the application and the specifications of the application. We outline the overall objectives for the application and the risks and opportunities identified.
- The application framework - This section looks at the various components used within the application environment
- Testing - This section shows how we chose to test the application to uncover any problems early and often, including a discussion of the use of Test Driven Development.
- Design decisions and structure - This section looks at the decisions made concerning front-end and back-end structuring of the application.
- Web security - This section shows decisions made around ensuring security for the application.
- Sprint retrospectives - This section outlines progress according to each of the two-week sprints, including any difficulties and where we decided to take a different approach.
- Reflection on the project as a whole - This section summarises the outcomes and achievements of the project and highlights any work which could be completed as part of a continuation of this project in the future.

Client Introduction

This project has been completed in collaboration with the environmental activist group Extinction Rebellion (XR). Originally founded in the UK in May 2018, XR now operates on a global scale with hundreds of groups operating in dozens of countries around the world (2020).

XR was founded in response to the mounting scientific evidence that the earth is facing a ‘sixth mass extinction’ and that, without a major shift to a decarbonised society, there will be ‘disastrous consequences for people and for all life on Earth.’ They are a non-violent organisation, utilising civil disobedience in order to draw attention to the current and potential future impacts of climate change.

A decentralised organisation, XR recruits volunteers from all walks of life who can plan and perform actions for the network as long as they agree to and adhere to the same principles and values. They have several different focused networks within the group collective which perform actions on media and messaging, finance and fundraising, training, art and others. They also have a group of dedicated technical volunteers who we have been in regular contact with while developing the project.

Our connection to the organisation is Carl Hughes, who acts as the client for our project. Carl is a freelance web developer who works part-time for XR on a voluntary basis, founding the aforementioned group of technical volunteers. The tech group meets over Skype on a weekly basis to discuss and plan work on the organisation’s technical resources. Their projects so far have included the completion of national and local websites for XR which were created entirely by the group with no outside help.

Our relationship with the client has been to act as temporary technical volunteers for XR (discussed further below). We are creating the application so that it can be made

available to the long-standing technical volunteers to adapt or build on as required in the future.

Systems thinking in developing our application

In order to complete this project successfully, we need to understand how our application will be used and how it will impact on other systems and processes within XR. A systems-focused approach is in direct alignment with XR's understanding of how many different interlocking systems affect something as globally significant as climate change. When developing the application, we can gain a better understanding of its true significance within the XR collective by undertaking systems analysis.

When we create a new piece of software for any organisation, we are aiming to change and alter their working practices. Although it could be argued that this redefinition of working practices goes hand in hand with XR's 'shared vision of change' (as described on their website), it is important to address both the social and technical impacts of what we are proposing to develop. Even where there is unlikely to be a resistance to change, there may still be a risk of nonacceptance if these factors are not considered.

XR's global reach and the many thousands of volunteers who engage with them demands creativity in day-to-day operations. Because they do not have a hierarchical structure, most decisions on behalf of the organisation are made at a local level. Meetings are held regularly to welcome new members, deliver training and plan actions. Then there are working groups focused on different topics, such as finance, media messaging, communities and tech (Extinction Rebellion website, 2020).

Within these different groups, each individual has different responsibilities. There are also representatives of each group who co-ordinate communications and feed back to to the wider organisation about current developments. XR also accommodates 'affinity groups', which are collections of people who may form in response to direct action (such as a protest event) but may be more transient in nature than the core XR groups.

XR's tech volunteer team works in response to the myriad challenges of connecting the organisation. From utilising crowd-funding websites to source money to choosing appropriate instant messaging applications so that volunteers can stay in regular contact with each other, XR effectively exploits communication tech to support its decentralised structure. The tech group works to produce branded digital content for the organisation, such as its official websites. The development of our application can be seen to take place as an extra arm of the XR tech team on a temporary basis.

If we consider a system to be a 'set of elements so interconnected as to aid in driving toward a defined goal' (Gibson et al, 2016), then Extinction Rebellion offers us a clear example of systems thinking. The disparate groups of its volunteers (known as 'rebels') based all around the world are joined together in support of a common goal; to protect the planet and all of its inhabitants. Within this large-scale system, smaller sub-systems operate at a local or issue-based level with representatives feeding back information.

The tech group we are working with operates on a global basis, with the majority of its members based in the UK. Two of the members of the group are responsible for the production of the official websites, one of whom (Carl Hughes) representing our contact for the development of this application. There are no formal systems in place for development of tech solutions in XR. Much of the work is driven by ideas formed by the tech team itself, informed by feedback from the activist groups.

The achievement of XR's goal demands a challenge to many systems, social, political and economic, many of which are highly impactful on our everyday lives. Climate change campaigns in general require the use of systems thinking to be successful. To solve the current situation, human beings need 'the ability to stand back and see the big picture, and look at things in transdisciplinary ways.' (Ogden, 2019).

As such, XR takes a wide approach to campaigning, rarely putting forward specific solutions or fighting frontline issues (although they still support this form of activism) but rather favouring large-scale civil disobedience aimed at the seat of power. Their activities have included partial or full shutdown of public spaces, roadblocks and mass marches (Extinction Rebellion website, 2020). Traditional methods of environmental protest, such as letter-writing and petitions, are typically siloed in individual communities with no little to no focus on the global scale of the environmental crisis. XR's promotion of 'above ground' mass protests and civil disobedience, which is in full view of the public, raises awareness on a wide scale and promotes the idea of the movement as being participatory and connected.

Despite their global reach and heavy reliance on flexible and fast communications between groups, XR is not ignorant of the more negative impacts of some of the advances in the tech area. An article on their website entitled 'Smartphone: Friend or foe?' (2020) states that smartphones have been linked to 'anxiety and behavioural problems' as well as a 'growing disconnection to nature'. An XR-affiliated group in New York City known as 'Tech Action' (2020) explores how to challenge Big Tech's funding of climate change denial. In addition, given the flat hierarchical structure of the organisation and the fact that XR activists can represent many different ages and backgrounds, it is unlikely that there is only one opinion of the efficacy of tech solutions within the organisation.

Nonetheless, the majority of XR's systems and processes rely on communications technology with even the smallest organising groups having some kind of social media presence to organise their activities. Our aim is to ensure that our application is used by XR activists (and those interested in XR's activities), that it is easy to understand how to use the application and that we are opening up an avenue for individuals who may not otherwise have been able to access this information (those who do not currently have access to Facebook).

For a period of time, the developers within the volunteer tech team would co-ordinate development projects themselves, including accommodating feedback from XR activists who would request changes or adaptations to the digital solutions. However, the conflicts between the requests were too difficult for the developers to accommodate along with the demands of the work and so a co-ordinator was added to the team. The co-ordinator could gather information from local and national group meetings to feed back to the developers as well as prioritising tasks to be completed. Our connection to XR did not involve speaking with the co-ordinator - this was due to conflicts in availability.

The work of Extinction Rebellion is designed to focus on the bigger picture - to see how the efforts of many can link together to effect social change on a global level. Because there is a strong need for constant communication and dissemination of information, they have a strong base of digital solutions.

A major risk of our development process is failing to account for the full extent of the needs of those who will use it. As Cherns refers to in his theories on social-technical design, if system builders fail to account for the social requirements of the system being constructed, those needs would be met in “some other way”, likely impeding the organisation more than helping it (Siau et al, 2011). In the case of our application, this could be users deciding they do not want to use our application and consider it easier to only use Facebook. They could also choose to replicate the information manually to share on other websites or through email or instant message. This could lead to an exacerbation of the current issues which led to the need for the application in the first place (discussed further below).

To mitigate these risks, we will ensure that we take the following steps:

- Leverage Carl as a promoter of our work within the organisation to convince activists within XR to use it
- Produce a clear and concise user manual to create a shared understanding of what the application does
- Seek feedback on the application's functionality ahead of completion where possible
- Keep a close focus on the requirements for the application at every stage of development

Defining the Usefulness and Importance for Our Application

The development of any software application is centred around solving problems. However, the design of the application is only possible when a problem has been identified and possible solutions considered (Dasso & Funes, 2007).

Because there is an existing volunteer team within XR which specifically works on technical solutions, we were fortunate that we had a source of ideas and requirements which had already been worked through. The group had several ideas which they had either put on the back-burner or had not yet found enough support to develop. One of these ideas was for a mobile application to update in real-time to provide live information for volunteers attending XR events. Another idea was for the Facebook events-based application which we decided to choose as a project.

The reasons for choosing the events-based application for our project were as follows:

- It is an application with an obvious and immediate use-case
- It has a relatively simple concept at its core but can be built-out to incorporate other features as necessary, including more sophisticated business logic layers
- Following this, the various components of the application can be easily broken up into different sections for two people to work on simultaneously without high dependence

Even if the original idea for this solution had not existed, we can easily ascertain the requirements by looking at the current heavy reliance on Facebook to advertise events within XR. Although XR has shifted away from using Facebook-owned WhatsApp for most of its communications (preferring to use less well-known applications such as Telegram and Signal which are seen as more secure), Facebook is still commonly used within the organisation. Members set up pages to share and advertise information as

well as groups which may be public or private. Events and activities are more commonly listed on Facebook than anywhere else.

Extinction Rebellion identified the need for the unification of their events information due to two main factors; (1) the high-speed growth of the organisation was making it very difficult to quickly locate information about all current events and (2) it was becoming increasingly difficult to quickly update event attendees about last-minute changes as there were too many different places where events information was being entered.

As part of our initial conversations with the client, we discovered it was becoming difficult for prospective XR volunteers to easily get a picture of what was happening in their local area and receive updates efficiently. There was an additional problem in that organising groups have a heavy reliance on Facebook which is inconvenient (not to mention potentially alienating) for individuals who do not have Facebook accounts. Finally, it was difficult to maintain consistency about how events were publicised and information shared. Individuals volunteering within XR had previously written code for local XR group websites to pull events from Facebook however the methods used to do this could be slow and inconsistent across the sites. Having one place for event information to appear would remove the need for checking multiple websites or Facebook pages while allowing flexibility for how events data is updated and edited as necessary.

Project Development Process

When choosing a software development process, there are some key points to consider concerning the constraints of the project. The development process for any software project is shaped by a number of factors, including:

- The level of flexibility available in the requirements
- The definition of the end-users
- The scale and scope of the project team
- The timeline for the project

A traditional Waterfall process model would require a single pass through each stage of development (from requirements gathering to design through to implementation) (Gilb, 2007). Each stage of the Waterfall process is well-defined, which makes it ideal for projects with a fixed timeframe and fixed deliverables which (to an extent) apply here. However, an Agile process provides more scope to adapt to changing requirements and puts more emphasis on quick production (Agile manifesto, 2001). There are pros and cons to each approach which are discussed below.

Flexibility

We have a degree of flexibility within the specification of the application in that there are some core requirements (eg displaying all event data from Facebook in one place) and optional extra requirements (eg adding in campaign information to each event). As the Extinction Rebellion development group meets weekly, there may be additional changes or requirements which will become apparent over time. Therefore, an iterative process such as Agile would be more suitable than a process such as Waterfall where requirements need to be more rigid.

End-users

The target end-users for the application are widely dispersed, rather than tightly controlled. Although we are mostly working with one representative for the client, there may be times where we need to incorporate feedback from other XR members quickly. For this reason, it would not be a good idea to use a Waterfall method of development (where all requirements are fixed at the beginning and feedback is only sought at the end of development).

Scale and scope

There are two developers on the team and therefore there is less need for elaborate project plans to coordinate work (as in the Waterfall method of development). Documentation is still required for the project, however it should be kept clear and easy to understand.

Timeline

Rapid progression is needed for development on this project due to the relatively short timeframe for development. In this way, an Agile methodology for development would be the most suitable. However, there are additional constraints on the project which are that both developers are working on the project part-time in different locations. This is where a methodology like Waterfall, with rigid project management tools, may be more appropriate. Overall, weighing this concern against the need for flexibility and quick adaptation, we chose an Agile methodology for development over Waterfall.

Agile Project Management

There are lots of different understandings of what Agile development is and how it works. The original Agile manifesto, published in 2001, was created as a way to push back against excessive planning and documentation which was slowing down software development for most businesses. It contains values which emphasise the need to create working software quickly, rapidly respond to change during development and regularly communicate with the customer rather than wait until the end of development.

One of the ways in which Agile can be a good fit for part-time student projects is the definition of individual tasks as user stories which helps to ease the planning and dissemination of work. If stories are well-defined and require minimal amounts of code with good test coverage, then any development team can successfully divide tasks even if they are not necessarily working at the same time (Kiely, 2019).

Understanding the constraints of working on a part-time, student project meant that we were able to make a decision about a suitable piece of work side-by-side with the client. The client already has a development team who work as volunteers and therefore there is a lot of flexibility about what we can work to deliver for them.

In initial discussions, the client had several needs for development, including a centralised event application and also a messaging system to allow volunteers and other attendees to receive live updates during demonstrations. We chose to develop a centralised event application as there was a more obvious path to step-by-step development and the requirements would fit more comfortably with the skills and experience we have both gained as part of the degree course. There was also a degree of flexibility in terms of additional functionalities which could be added to an application of this nature. This could not only make it a more attractive and useful resource for the client but provide a more in-depth showcase of our learning experience.

One of the core practices of Agile, as implemented, involves working incrementally - breaking up a large problem into a series of small problems that can be independently verified (Douglass, 2016). Modern Agile software development often involves breaking work into cycles known as ‘sprints’, which are usually 2 weeks in length, using ‘user stories’ to define units of work that can be completed in the sprint and utilising methods of tracking the progress of each story from ‘to do’, to ‘in progress’ to ‘complete’.

Sprints involve several stages, including planning meetings to organise work, daily stand-up meetings to discuss the previous day’s work and raise any roadblocks and reviews where the work completed within each sprint is discussed. There is also a sprint retrospective where work and team dynamics can be reviewed in a more general sense.

Within the dissertation project, there are several constraints which mean that standard Agile working practices have needed to be adapted for our needs. Because both developers work in full-time jobs, it is obviously impractical to hold daily stand-up sessions. In addition, while it is not often difficult to find at least one evening a week when both developers are free to meet through Skype, this meeting may not always fall on the same day each time. On weeks where we may need to meet with the client to discuss progress, it is also a challenge to then also fit in an extra evening meeting just for the developers.

We have been able to mitigate these challenges by ensuring regular Skype meetings are scheduled once a week between developers, once a month with the course supervisor and once a month with the client. The Skype meetings have incorporated aspects of a stand-up session, where we have been able to discuss progress and roadblocks, and also sprint planning sessions, where we can check where we are in a sprint process and plan work accordingly.

As a lot can progress within a week, we have also used WhatsApp to communicate between ourselves as developers and also with Carl Hughes as the client. This was especially important given that it was not always possible to know when the other person was going to be working on the project. It was vital to use messaging to tell each other about what we had worked on in a particular evening and share any difficulties or useful resources.

As well as the development teams involved in Agile working, there are also several roles which act as co-ordinators such as a team lead to assign work, a Scrum master to lead Agile working practices and a product owner who represents the client to the team.

Defining roles within the Agile process happened fairly naturally as Kit had more of an affinity with front-end development while Sarah was more comfortable working with back-end structure and data migration. A challenge to overcome was the division of responsibility concerning oversight of the entire project - scheduling meetings, ensuring that work is being completed to schedule and co-ordinating tasks. In a larger Agile team, a role of Scrum Master or Product Owner could have been assigned to another member of the team so that work could be more focused on development.

However, the fact that there was a clear common goal for the project (with high levels of authority as well as responsibility for both team members) allowed for the team to be successfully self-directed; the benefits of this likely outweighing the cost of having no formal project oversight. According to a survey with data gathered from over 500 organisations, self-directed teams have the benefit of greater flexibility, faster response to technological change and improved quality of service (Williams, 1995); all of which is directly relevant to a successful Agile project.

Developing Requirements with an Agile Approach

If we were working in a traditional Waterfall development methodology, we would need to fully understand and vet all requirements for software development prior to project initiation. All development would then take place based on this initial agreement of requirements. For us, an Agile approach to requirements gathering is one which takes into account that change (and possibly disruption) will occur during development, that clear communication with the client is required from start to finish and that developing features is more important than extensive documentation. In order to adhere to Agile practices within our development process, we will do the following:

- Avoid being too tightly bound to the details of the technical development until the work has begun. This is especially helpful for a student project where both participants have additional responsibilities - there will likely be setbacks and other disruptions.

- Form each requirement so that a feature can be developed as part of a two-week sprint, allowing our efforts to be focused on only one part of the application at a time. This also allows development to be more streamlined and time is less likely to be wasted on things which are unnecessary for the application.

- Ensure regular communication with the client to avoid any disconnection between initial agreements and what is eventually delivered. It is important to acknowledge that change will happen and that we must adapt to this rather than sticking too rigidly to a plan. (Software Project Planning & Control, 2013).

Part of Agile requirement discovery is the development of user stories. A user story is a requirement that fits into a simple template, describing a user's role, what they want from the application and what purpose the feature will serve for their role (Vanderjack,

2015). This allows each requirement to have the context of what the client is asking for, explicitly spelled out and with clear information of the value-add for the organisation.

Stories are kept short and concise so that they can be easily translated into technical requirements for development. Once the technical requirements have been ascertained based on the user stories, an estimation is given to each one, sometimes using what is known as ‘story points’. The story points reflect how long it will take for the feature to be developed, based on the viewpoint of the development team. The story points are not based purely on time estimates; they should also take into account the perceived difficulty of each task and also the accuracy of estimations from past work (Radigan, 2020).

The requirements then form a ‘backlog’ which is essentially a to-do list for the application features to be developed and delivered in sequence. The requirements can be refined during the development phase as frequent communication between the client and development team is essential (Measey et al, 2015)

Adapting Agile requirements gathering for our project

There are a number of ways in which we have needed to adapt traditional Agile processes for gathering and refining requirements in our project.

Any development team working in an Agile framework will have difficulty estimating their velocity (how fast they can complete tasks) at the beginning of a project. Without prior estimates to refer to (as well as whether those estimates were correct), there is insufficient information to support how quickly a feature can be developed (Brizard, 2015). This issue is magnified in a small student project where the developers’ skills are still being strengthened and team members need to fulfill several different roles at the same time.

Having the freedom to keep requirements open has helped to mitigate this problem, along with allowing flexibility between ‘sprints’. Each sprint period of two weeks acts as more of a guideline than a strict deadline - work can be pushed into the next period of time if necessary as a response to implementation difficulties. The sprint periods act as a useful way to monitor progress of work and ensure we are on track to deliver the requirements.

Another adaptation we have needed to make is considering that the requirements are a representation of a *potential* user’s needs by proxy. This is because we worked with one person to develop the requirements for the project through Skype meetings, rather than user groups. A risk of this approach is that requirements will be skewed towards one person’s opinion rather than fully representative of how the application will be used.

This risk has been mitigated in several ways: (1) the person representing the client is also a volunteer within XR. They therefore have an understanding of how the application should work in practice as they would be using it themselves. (2) we can understand that our application could potentially be used by anyone, either existing volunteers with XR or individuals who are interested to attend events. We can therefore utilise design and functionality assumptions based on best practice as well as the given requirements.

Requirements gathering process

A requirements-gathering process is designed to do two things: (1) provide a structure for what the client wants from the application and (2) build a shared understanding between the development team and the stakeholders for the project (University of Minnesota, 2020).

In order to gather the requirements for our application, we held a conversation with Carl over Skype to get an idea of what exactly the application would ideally do. We provided an opportunity to discuss the different aspects of the application in order to flesh out a set of initial requirements with the understanding that this would be amended throughout the development process.

With an Agile approach, it is important to have this conversation face-to-face rather than rely on communicating requirements through documentation. This is to avoid any miscommunication between the developers and the client. However, this approach did not mean that no documentation was produced at all. After the initial conversation, we provided a requirements document for Carl to confirm (included as an appendix).

This document included the goals for the application, a summary of why it is needed within the organisation and a list of user stories. The user stories were created as examples of what several different user groups would need to use the application for - these were divided between users (who could be existing volunteers with XR or members of the public who are interested to attend meetings) and administrators (people already working with XR who may need to access the backend of the application to change or update information).

Within our defined requirements, we can identify those which can be defined as ‘functional’ (such as being able to filter and sort events to display on the homepage) and ‘non-functional’ (such as ensuring that the application is secure and accessible to all users). The functional requirements can provide a basis for how we might design our tests (see example given in ‘Testing’ section) (Altexsoft, 2018).

Business requirements and UML

Following the requirements gathering phase, we can then detail the events and processes involved in how the application will be used, identifying the business rules and activities that need to be taken into account during development. This can form the business requirement for our application, while the specific software requirement is detailed in the section on requirements gathering (on page …) (Laporte & April, 2017).

We can use modelling to represent the business processes and capture the properties of the problem domain, without adding unnecessary detail (Open University, 2013). UML (Unified Modelling Language) is a popular standard for use in many different processes and practices to display the expected behaviour of a system (*what*) without showing the exact method of making it happen (*how*).

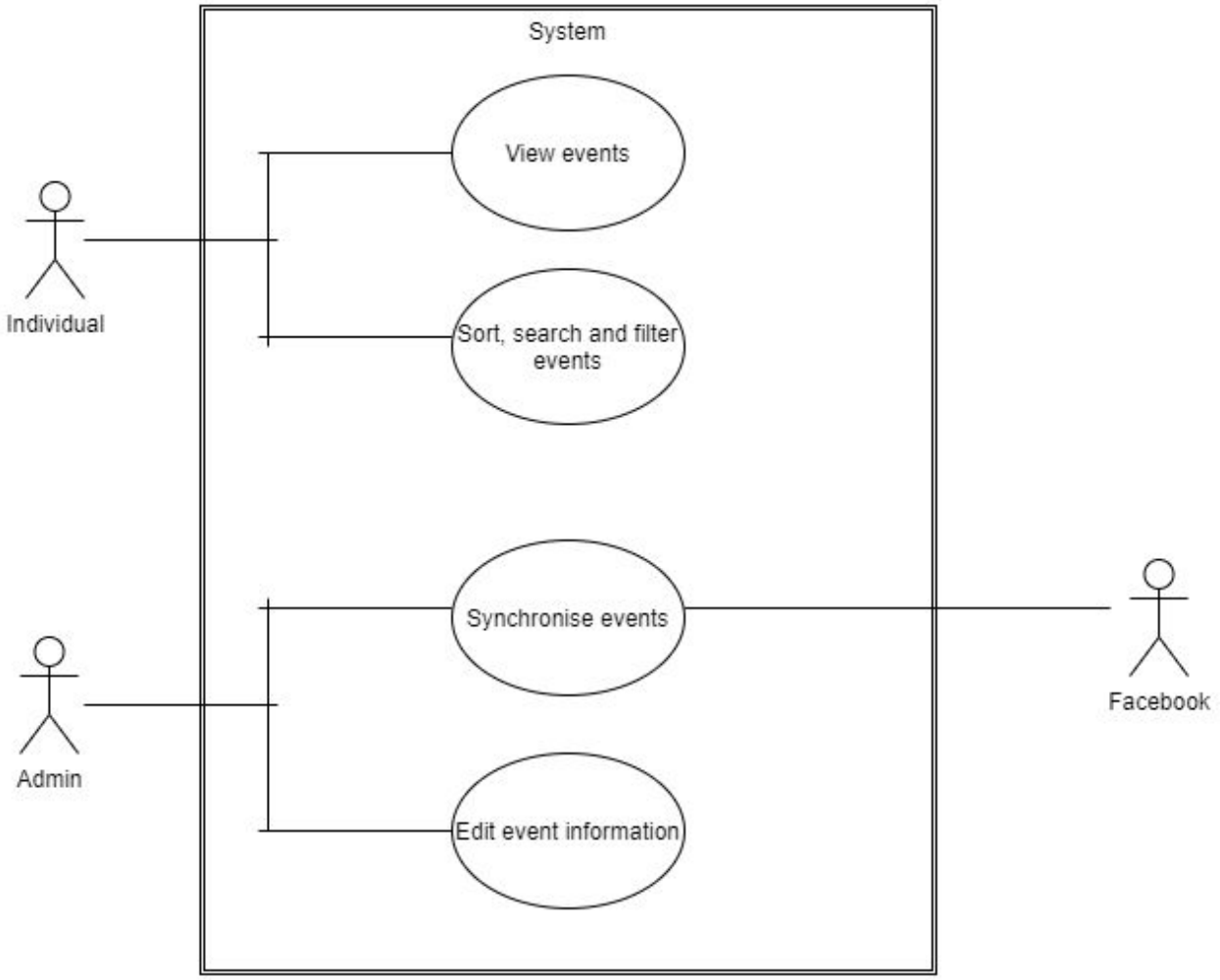

Figure 1: Use Case Diagram

There are several different kinds of UML diagrams which can help us to display the behaviour of our application, including use case diagrams, sequence diagrams and activity diagrams. The use case diagram (Figure 1) above shows how our application should work in practice with representation of actors (individual users), use cases, relationships and the system boundaries.

A use case diagram can help us to define the practical usage of our application in an easy-to-understand way, so that we can communicate with our client effectively. We can also link the key development activities to the desired behaviour of those interacting

with our system, fulfilling our goals and objectives from a user perspective (Ambler, 2004).

User stories

- *As a user I want to see events in a list so that I can understand which ones I might like to attend*
    - **Objective:** Collate and display information about XR's events from Facebook
        - Information related to events will be displayed clearly and logically to the user
- *As a user I want to filter events by location, date and type of event so that I can find events which are the most relevant to me*
- *As a user I want to see recurring events listed in a logical way so that I can see all instances of the same event*
    - **Objective:** Allow information about events to be sorted, searched and filtered as relevant to the user's needs
        - Using the event information, such as location and date, we will create logic to allow users to filter, search for and sort the events information according to what they most want to see.
- *As a user I want to see events shown on a map so that I can easily find directions to events closest to me*
    - **Objective:** Allow information about events to be displayed geographically
        - Events information must be displayed on an interactive map where users can clearly identify events which are close to their home location.
- *As an administrator I want to login to the app and change event details so that I can have flexibility and control around keeping events up to date*
    - **Objective:** Ensure there is capacity to regularly synchronise the information from Facebook
        - Information related to events will be up to date at all times

- **Optional:** *As a volunteer I want to access an RSS feed so that I can easily display centralised data about all events on my local XR website*
    - **Objective:** *Provide an accessible RSS feed of events*
        - The ability for websites representing smaller XR groups to access an RSS feed of events would increase the usefulness of the application.
- *As an administrator I want to be able to change default images on each event so that I can customise information as necessary*
    - **Objective:** Feature administrative controls
        - Users must be able to login in order to edit or remove information relating to the events.
    - **Objective:** *Allow an administrator to replace or remove images pulled from Facebook*
        - This would allow for additional flexibility in the display of information relating to each event.
- **Optional:** *As a user I want to see relevant social media posts on each event so that I can see how events link to national XR activities*
    - **Objective:** *Provide information from social media as linked to each event*
        - If an event is linked to a particular campaign or national event, information from social media could be displayed alongside the event details to provide more context to the user.
- *As a user I need the application to be fully accessible so that I do not face access barriers due to a disability or impairment*
    - **Objective:** Have an accessible user interface
        - The application must be navigable by keyboard and adhere to accessibility standards.
- *As a user (or volunteer or administrator) I need the application to be secure so that I can feel confident that the application will not be at risk of defacement or infiltration by hackers*

- **Objective:** Have appropriate security measures included in the design and development
    - The application must include measures to tackle security concerns such as cyber-attacks or the display of unauthorised events.

Story mapping for our project

Agile methodology focuses on breaking down requirements into stories which can be developed (mostly) independently from one another. This helps to reduce the amount of potential bottlenecks and also allows for easier incremental development, so that working software can be released to the client sooner. However, it is still important to consider the product as a whole. The risk of focusing only on large features is that the smaller, operational work will be forgotten and the initial iterations of the product will be less useful as a result (Patton, 2005).

Our development process is different from the traditional Agile approach as we are having to allow extra time for our individual learning processes (more things are likely to go wrong as we are both students so it would not be helpful to be too rigid about project deliverables during development). Being able to work in two-week sprints was still helpful throughout the project as it provided a structure and a focus for our work. It helped us to build up the project logically, moving from the groundwork of setting up the application to communicating with Facebook's API to providing logic to display and filter events. Story mapping gives another view of this process.

Patton's approach of breaking up user stories into activities, tasks and sub-tasks can be applied to our backlog to provide context to both the amount of work required and also the logical order tasks should be completed in. For example, the first user story listed (*As a user I want to see events in a list so that I can understand which ones I might like to attend*) is required to be completed before the user stories concerning events being

filtered or sorted. It also requires us to set up and verify the call to Facebook's API which is likely to take the most amount of work.

Even for a small project, it is useful to sort the small features according to priority and user behaviour for coherence and clarity as the project progresses. Below is a story map (Figure 2) of our user stories with a breakdown by user role:

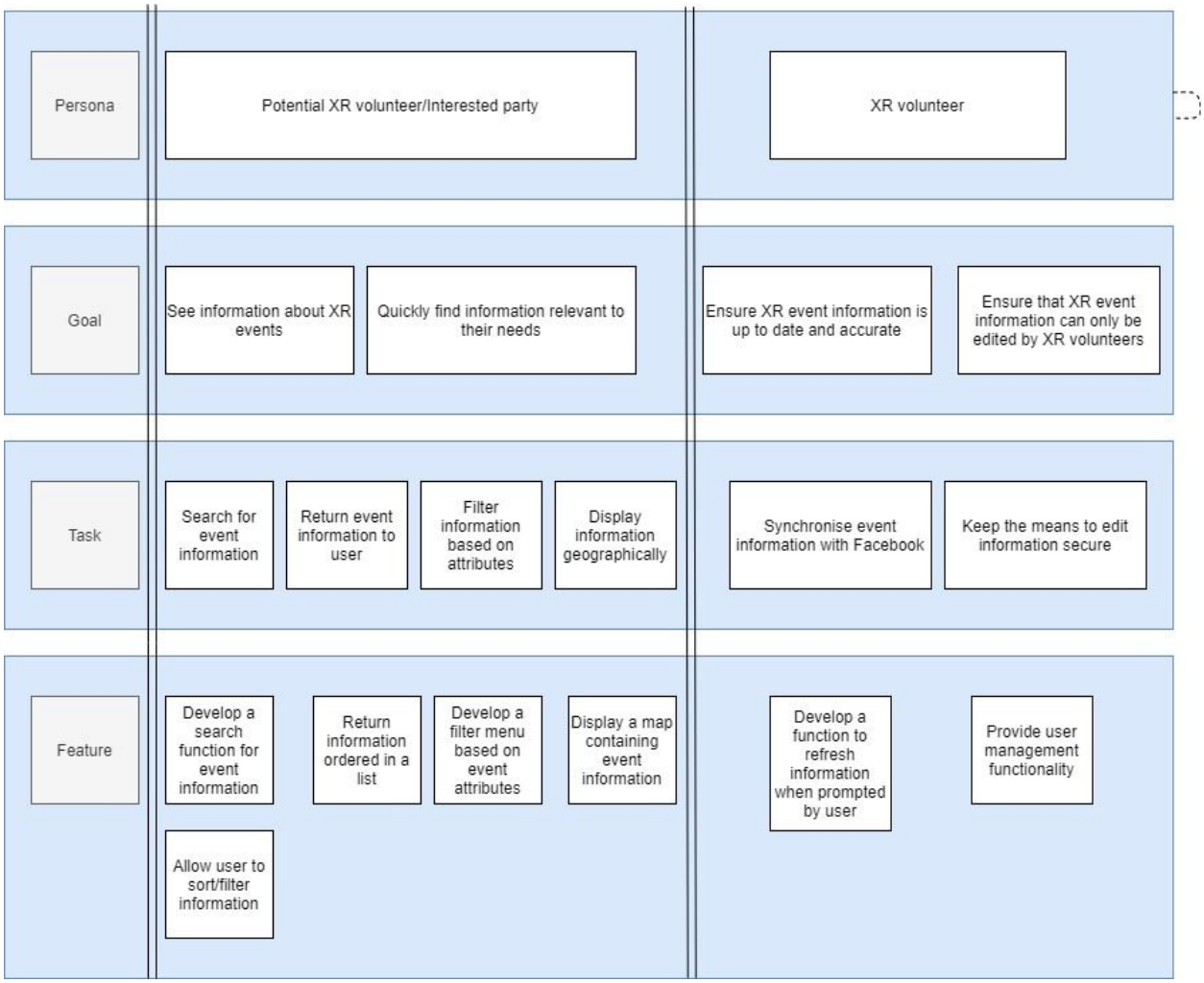

Figure 2: Story map

Risks and opportunities

As mentioned earlier, every software development project carries risks, some which are common and to be expected and others which are more specific to the individual project. Some of the most common areas of risk in any development project include misunderstanding of requirements, changes to requirements and a lack of an effective project management methodology (Arnuphaptrairong, 2011). We can mitigate these risks with the choices that we make during development, however it is important for us to identify the most likely risks specific to our project first.

Effective risk management will reduce the chance of project failure, identify project elements which require special attention and provide alternative strategies at an early stage. Some examples of risks which are specific to our project include:

- Unknown unknowns - It is difficult to identify things which have not happened yet and the development team does not have a huge amount of past experience to draw on in order to predict what may go wrong
- Tight timeframe - Because the development team are working on this project while working full-time, there is less flexibility to fix problems and devise alternative courses of action if things go wrong
- Differentiating responsibilities - The development team consists of only two people which increases the risk of a single point of failure or confusion about responsibilities

It is important to consider that, with every risk, there is also a chance for opportunity. The benefits of pursuing an opportunity could outweigh the impact of the identified risk (Ruhe et al, 2014).

We can also assess the more common risks concerning requirements, cost, quality and operations as part of a wider risk management plan. Risk management plans typically comprise a description of the identified risks, their different classifications, an analysis of

how likely they are to happen and an outline of how each risk will be mitigated (Software Project Management & Product Control, 2013).

An example of how we can classify, prioritise and mitigate the risks of our project is discussed in Table 1 below.

Risk	Probability	Mitigation Strategy
Timeline may be too tight to allow all development features to be implemented	High	● Regular meetings with Carl to track progress ● Ensure any changes to requirements are regularly assessed ● Being realistic about development, ensuring MVP is delivered
Taking on development tasks that are too complicated and cannot be completed in time	Medium	● Focusing on implementing one feature at a time ● Sticking to two-week sprints to track progress closely ● Ensuring that must-haves are prioritised over nice-to-haves ● Keeping progress

		logs updated and having regular meetings between developers to support each other ● Ensuring regular meetings with supervising lecturer
Application could be targeted by hackers due to political nature of HR	High	● Ensuring that adequate protection is implemented in the code (eg protecting against SQL injection attacks)
Both developers work full-time - development could be sporadic and difficult to maintain flow	Medium	● Ensuring regular weekly meetings are kept between developers ● Communication is available often through WhatsApp

Table 1: Risks and opportunities

Application Development

Front-end Framework

The front end is a term that means client side programming. Front end scripts are client side scripting that is executed by a browser and the backend scripts, i.e. server-side scripting is executed by a web server. (Mayers, 2017). For this, we chose to use the Bootstrap framework. Bootstrap is a well known and open source CSS framework that was created by developers from Twitter. The framework contains component and design templates that cover a range of features like typography, forms, buttons etc.

Bootstrap has one of the best responsive, mobile-first grid systems available. It is built with Flexbox. It helps in scaling a single website design from the smallest mobile device to high-definition displays, logically dividing the screen into 12 columns, so that you can decide how much screen real estate each element of your design should take up.

The main reason why we chose Bootstrap is because of ease of use. It is very easy to set up and a functional website layout can be created within an hour. Code snippets and templates from their online developer community and documentation are everywhere thanks to its popularity and friendly support.

This allows us to save valuable time as we do not need to code everything from scratch. Bootstrap's predefined design templates and classes allow developers to display components exactly where they fit. If you have the basic working knowledge of HTML and CSS, you can start development with Bootstrap.

Back-end Framework

When choosing a programming language, one factor influencing the decision was the focus on Ruby on Rails in several modules of the MSc IT course. Given that both

developers were required to have previously submitted coursework which relied upon successfully working Ruby code, this would seem an obvious choice. However, this is not the only reason that Ruby (and Rails) is the most suitable language choice for this application.

As an object-oriented language, the syntax of Ruby fits the way that developers perceive and think about the world (Ruby for Beginners, 2020). Real-world concepts such as ‘users’ and ‘items’ are simplified in code as ‘classes’ and ‘objects’. Ruby’s reasonably natural syntax makes for a big advantage for beginners to web development. Its scripting is simple and clean and it is not difficult to work out what is happening within any piece of code. While developing the application, this will make the component parts easier to create and also to maintain.

As a development framework for Ruby on Rails allows developers to write less code while accomplishing much more than is possible in other languages (Rails Guides, 2020). One of the core facets of Ruby on Rails programming is convention over configuration. A developer does not have to spend lots of time adapting files in order to get a project up and running. This is especially helpful given the relatively short time for the project to be developed.

There is an emphasis on RESTful application design within Ruby on Rails which is particularly helpful for the kind of application we want to develop. RESTful applications are a type of software architecture which is based on the client-server relationship. By separating the user interface from the data storage level, the user interface can be portable across many different platforms. The design takes advantage of existing protocols, meaning that developers do not need to install libraries or any additional software to see the benefits (Hartl, 2020).

Ruby on Rails allows for fast and responsive development which is ideal when used in conjunction with an Agile project methodology. Agile development allows for multiple and rapid changes in the requirements of the project; Ruby on Rails allows the process of programming to be much faster than other languages, partly due to a large selection of open source code available. The open source code and libraries (known as gems) also make it easy for applications to be customised and modified as needed. To make our application unique and fit for purpose in a short space of time, Ruby on Rails provides more than enough material.

Another part of Ruby's simple syntax is the concept of Don't Repeat Yourself (DRY). This means that there is no repetitive code within the application which makes maintenance easier over time. It also leads to less ambiguous code use which is preferable when developing in a team. Keeping the principle of DRY in mind during development is key to ensuring a reduction in quick-fix solutions implemented in a hurry close to deadlines - an obvious risk in a project on a tight timeframe (Peters, 2012).

Ruby on Rails also has a strong focus on testing and is ideal for Test Driven Development (TDD), discussed later in this dissertation.

Integrated Development Environment

For the application development, we decided to go with an IDE called Rubymine. An integrated development environment (IDE) is a software suite that consolidates the basic tools developers need to write and test software.

Typically, an IDE contains a code editor, a compiler or interpreter and a debugger that the developer accesses through a single graphical user interface (GUI). An IDE may be a standalone application, or it may be included as part of one or more existing and compatible applications.

The big benefit of an IDE is the “integrated” part. You do not have to switch between different tools to design a layout, write the code, debug, build, etc. You can do it all in one place. Because it is an integrative tool, these parts are designed to work well together. This is especially true for us since we are coding between WIndows and Mac.

We were fortunate enough to receive a Github Education pack after signing up. Rubymine was one of the numerous packages that was given to for free in a 2 year duration. Rubymine allows us to code on different platforms like Linux, Mac or Windows.

PaaS Platform

For the application platform, we have decided to go with Heroku. Heroku is a platform as a service (PaaS) provider that companies or developers can use for deploying and hosting applications in the cloud. Orlando (2011) states that “the defining factor that makes PaaS unique is that it lets developers build and deploy web applications on a hosted infrastructure”. Essentially, it handles all the infrastructure, allowing us to focus on our application to try and make the most of our 6 month deadline.

As a developer, we can deploy from Git again by simply doing a push to the respective server, and the appropriate frameworks are installed or updated in the cloud. We can also perform remote commands for database migrations and other tasks. Essentially, we chose Heroku for the following reasons:

- Speed: Heroku is fast, coupled with Node.js and Angular, it often creates apps that load in less than half of a second.
- Security: Heroku uses next-level technology to prevent hackers from stealing data or defacing your website. Heroku's infrastructure is PCI-1 Compliant.

- Simplicity: It requires very little configuration and setup to launch websites. It's also very easy to sync code and data between the developer's machine and the production environment.
- Integration: Heroku provides developers with a list of integrations: Heroku Postgres, MongoLab, Papertail, Slackin, Telescope and ContentFocus CMS are just a few of the hundreds of options available. Having these options allows us to choose the right integration to quickly customize any application.
- Scalability: Apps operate in smart containers, known as dynos. Dynos are lightweight Linux containers that run a single user-specified command. For our project, one dyno is plenty to run our application.
- Uptime: Heroku promises 99.99% uptime. Transactions are continually archived to a geographically distributed data store while automated health-checks are performed every 30 seconds to make sure the database is available.

Application Architecture

Because we are using Ruby on Rails for our application, it is helpful for us to consider the key benefits of Object-Oriented Programming (OOP) when looking at code design. OOP defines ‘objects’ in programming as combinations of variables, functions and data structures (Mulonda, 2018).

Our application is relatively small, therefore we only have a few objects that we are configuring for basic functionality. The ‘Events Controller’ class holds all of the methods which instantiate the business logic of the ‘event’ object. This includes the logic for the search, sort and filter functions.

Ruby also allows us to use what is known in OOP as abstraction, hiding the more complex details of what is happening in the program so that the interface can be viewed as easily as possible (Martin, 2018). For example, the use of ActiveRecord to access data in our application is an abstraction of SQL.

Abstraction is also a key part of how Ruby on Rails applications utilise DRY code (code which is not heavily duplicated through the application - Don’t Repeat Yourself). We have tried to minimise duplication of code wherever possible to make the application easy to maintain, however there were some places where it was not possible to avoid (such as the load_event_by_id method in both the Home and Event controllers). In a project like ours where we are learning code as well as implementing it, there is a higher risk of incorrect abstractions being implemented if we choose to always abstract rather than duplicate. Sandi Metz explains this issue as “prefer[ring] duplication over the wrong abstraction” (2016).

Inheritance is another facet of Object-Oriented Programming which could have helped us if we were designing the application with a much larger view. For example, if we

decided to acknowledge multiple different event types (such as online events vs. face-to-face events), we could have multiple subclasses which inherit from one overarching Event class. However, within the time restrictions and multiple roles being performed by two developers, this would be a task to be completed in a future design of the application.

The balance between ensuring future-proofing for our application while also ensuring we are delivering working code today has been a challenge. Metz (2019) acknowledges “In a small application, poor design is survivable. [...] However, the problem with poorly-designed *small* applications is that if they are successful they grow up to be poorly-designed *big* applications.” However, it can be stated that we have implemented solutions which promote OOP standards, such as cohesion (for example, separating the events data pulled from Facebook and the manually-edited information into two models - Event and Editevent - which can be adapted separately) and kept the design of the application simple (for example, not over-engineering the solution with too many classes and methods) for understandability and adaptability.

MVC

Rails utilises the Model-View-Controller architectural pattern, which advocates the separation of the data (model), the action direction (controller) and the user interface (view). The combination of these layers is handled completely by Rails, which means that our application can be developed rapidly. As long as we put the code in the right place and use appropriate naming conventions, everything should work (Gamble et al, 2013).

In addition, the model, view and controller components all act as separate entities. A change in our model should not affect a view and vice versa, helping us to scale and maintain the application more easily.

In our application, the control flow of the MVC model will work as follows:

- A user will interact with the interface and trigger an action (eg sorting and filtering the events list)
- The controller will receive the user input (eg the selected filters for the event list)
- The controller will assess the model (eg the database where the events information is stored)
- The controller will invoke a view which will render an updated interface (eg an updated events list)

Model

In our model layer (which represents the database), we have a main model called Event which holds data regarding all of our XR events. This is where we are storing information for use in the event views, however we are not in need of any front-facing logic to edit, create or update this data. The events table in the model is populated using our connection to the Facebook API.

Using a Ruby library which connects to Facebook's API called Koalas, we were able to extract data about XR's events. The data is returned in JSON format which then needed to be streamed into the backend database. Initially this was set up using Sqlite, however we decided to switch to using PostgreSQL. This was largely due to Sqlite's limitations in user management as well as compatibility problems when deploying the application in Heroku. PostgreSQL would allow future developers to set access privileges for users and administrators, making the application more extendable (Wang, 2017). The logic for how we used the data to populate the database consistently is described in the *Controller* section below.

We also have a model called Eventedits, which stores data which users in an administrative capacity have entered to update the events. We wanted to use a separate model (and therefore a separate table in the database) for this data due to the logic required to display updated information from two different sources: Facebook and user input. Because our application will be the single source of truth for events information within XR, if an administrator updates information on a single event, we do not want this update to be overwritten by the next Facebook synchronisation. Therefore, all manually inputted data is kept separate from the Facebook API data and the controller and views contain the logic for deciding what to display to the user (detailed below).

Controller

Generally, controllers will accept user requests, perform any necessary processing and then allow the view layer to display the results. Within our application, it is the controller layer which provides the logic for which events to display to the user based on their menu selections. However, we are also using the controller as a means to connect with Facebook's API and direct the flow of data into the database in a logical way.

Our synch_all_events method makes a call to Facebook's Graph API using the koala gem. We use the code in the Event controller to create new events where none already exist in the database, filling in the event title, description and details of the event location. If an event already exists in the database, there is logic in the code to update the database entry rather than create a duplicate event.

Views

The homepage and events page views display the events data via a combination of the Event model and the Editevent model. If an individual event has been manually edited, the view will display data from the Editevent model. If no data has been manually

edited, the view will display only the data from the Event model (which has been filled straight from the Facebook API). This logic is handled within the controllers as, in Rails, any direct interaction with the model should be decoupled from the view.

Business Logic

Designing the means to filter and search through the events displayed on the application requires extrapolating designing interactions which are most appropriate for what the end user requires. In the four main approaches to interaction design outlined by Saffer (2010), we are likely focusing more on user-centred design (where we are translating exactly what the user wants into design) and activity-centred design (where we are focusing more on the behaviour of users rather than their needs). Systems design (where we would focus more on the context of the computers, objects and devices within the use of the application) may be too advanced for our project and genius design (where the ideas of an experienced designer take a central focus) is not appropriate for our experience level as students.

We have additional challenges in that (1) we are designing an application where we are communicating with one client remotely with limited access to user feedback, and (2) we are required to make frequent trade-offs in design decisions due to our experience levels as web developers. The capacity for development and evaluation of alternative design solutions is also limited given the time constraints attached to the project. Mitigating these challenges meant ensuring frequent communication with the client (who made himself available through WhatsApp and frequent Skype calls) and also limiting rigidity on the type of interaction that we would like to implement. It would be far worse to fail to implement the core concern of the application than to complete a design which is simpler than originally envisioned.

Carl, in his role as our client representative, mostly took on the role of a proxy user - someone who could provide useful information about how the application would likely be used on a regular basis. He could also provide expectation management for us by reporting back to the XR development team on the scope and progress of our project and ensuring that the end product was not misrepresented. Ideally, we would have

involved end users in the development stages. This was difficult to arrange with the constraints on the project (both developers only working on the project on evenings and weekends as well as the additional challenges of Covid-19 meaning that fewer XR volunteers were available to provide support).

Filter menu design

The ability to select and display the events that a user most wants to see falls within the ‘instructing’ type of interaction. Within our application, we have a search bar where a user can type in a keyword to return a list of events which contain that keyword. We also allow events to be filtered using a dropdown menu. These navigational choices allow for quick, efficient interactions which can be easily repeated as necessary.

We chose to use a flat menu containing items which can be selected using dropdowns, appearing on the left-hand side of the screen. This allows users to see the full range of options to choose from and quickly select the ones most relevant to them. The benefit of using a menu interface as well as a search capacity also allows for users to recognise the options they want to see, rather than relying solely on recall. In other systems, where the full display of potential menu items may be much longer than ours, we may have needed to implement a different solution such as predictive text.

We used headings for the menu to break down the information attached to the events into simple categories. This makes it easier for users to scan through to select the options they would like. With a longer list, we could also have chosen to use an expanding menu which would allow more options to be shown on a single screen. However, expanding menus usually require precise mouse control, which would be at odds with designing the application to be accessed by a keyboard.

Responsiveness was also important. The menu selections needed to return results as quickly as the interaction allowed. This is to address the user concern that it is not worth having a separate application to hold events. If it takes too long to return results, they may simply return to Facebook to search for them there instead. Choosing to use dropdowns and a ‘search’ button also provides informative feedback to the user, fulfilling one of Shneiderman’s (2010) eight golden rules for system interaction development: for every user interaction, there should be clear system feedback, even for minor interactions.

One of the key guidelines for building the menu interface is to ensure that it could be easily changed (Lewis et al, 1994). An object-oriented programming environment, such as Ruby on Rails, is particularly suited to this. For example, ensuring that the size, colour or number of items were not hard coded and that the menu could be adapted to show different categories as needed.

The index terms used within the menu refer to the content within the events information. Choosing terms within existing content, such as place and event type, for the menu options provides context for the users to understand what they are choosing. In the case of ‘city’, the field already existed within the data pulled from Facebook. For something like ‘event type’, logic within the code extracted specific words from the event title (such as ‘talk’, ‘workshop’ etc). This also allowed for the event categories to be displayed in a consistent, predictable way (Morville, 2007).

Testing

Rails applications require a robust testing strategy to be in place from the beginning, as test code is included in the initialisation of all models and controllers (Rails Guides, 2020). These are not designed to test anything as such, however having the

frameworks in place made adding our own tests easier. In addition, Ruby has a number of testing libraries which can be utilised in order to write tests.

In terms of our specific project, the development challenges we faced were mostly centred on the short timescale, the small size of the team and factoring in time for learning as well as producing code. Writing tests helps us to meet these challenges as we ensure that the code produces expected results early, errors are captured and dealt with on an incremental basis and that each member of the team can see progress documented in the tests as they were written (Castello, 2018)

The above examples form a theory of development known as Test Driven Development (TDD), which was particularly relevant to the creation of our application in Ruby. As well as being a testing strategy, TDD is a wider development strategy that puts an emphasis on the behaviour of each part of the application before code is even written. It requires developers to focus on each increment of development and provides clarity on what has been correctly implemented and what remains undone (Edwards, 2003)

Given that we were working with a third-party API, there was an increased requirement for a specific guarantee about how the processes within the application should work. For example, an early test was written to check the synchronisation of Facebook events to the application. Being able to write an integration test before any code allowed us to ensure that the controller, model and third-party API were all capable of being connected correctly before any other development occurred.

One drawback to incorporating TDD within a student project was the sense of writing code which is not ‘doing’ anything alongside the large amount of code required for the assignment as a whole. This sense of overload, along with a desire to see an application ‘working’ quickly, can mean that it is tempting to skip writing tests in favour of simply writing the functions to close off a particular part of development. Because we

were working on a small project, we mostly focused on validation tests for the Event model as well as the aforementioned test for the synchronised data from Facebook. Nonetheless, taking the time to write tests helped us to understand more about the code itself as well as boost our confidence in the eventual finished application (Lemon, 2018).

Web Design Overview

According to Niederst Robbins (2007), web design is a process which includes a variety of different disciplines such as graphic design, information design, interface design, document production, scripting and programming, and multimedia. Website design is a crucial point in a website development process. It involves the arrangement of content into graphical models that can be used as a basis for coding a site (Almeida & Monteiro, 2017). Web design used to be focused on designing websites for desktop browsers; however, since the mid-2010s, design for mobile and tablet browsers has become ever-increasingly important. The idea of designing multiple versions of the same site to attend each size of screen and resolution is not possible because of how impractical it would be in terms of cost and maintenance effort.

Responsive design emerges as a technical solution to the above issue, since a website dynamically adapts to the width of the device in which it is being visualized. The content of a website should intelligently reshape itself for maximum usability and impact (Subic et al, 2014). A good web design is easy to use, aesthetically pleasing, and suits the user group and brand of the website. Many webpages are designed with a focus on simplicity, so that no extraneous information and functionality that might distract or confuse users appears.

For this project, we need to design a website that fulfills all the points mentioned in the above paragraph. We started the project off by creating a simple wireframe design on Google Slide. A wireframe is a visual representation of a user interface, stripped of any visual design or branding elements. It is used by web designers to define the hierarchy of items on a screen and communicate what the items on that page should be based on user needs (Mears, 2013). Essentially, wireframing is a “plan” for websites. It shows the web page layout by illustrating the positions of content and navigational elements, and also represents functionality.

Wireframes are not meant to be distracting. They are meant to create a conversation about the overall user experience. For this reason, we opted for no colors to be used in our wireframe, as seen in figure 3. The key to a good wireframe is simplicity. All we need realistically, is to show how elements are laid out on the page and how the site navigation should work.

Figure 3: Wireframe Design.

Wireframing is extremely important as it gives us an opportunity to focus more on the user experience of the website. It may seem counterintuitive to create a wireframe first rather than going straight to web development but it actually saves a lot of time through the entire project life cycle. It allows us to test and refine navigation and study the overall effectiveness of the page layout against usability best practices.

Wireframes enable both client and web designers to create a solid plan before committing to building the website. In the client's perspective, it allows them to focus on

the essential information and navigation elements rather than the style or use of colors. They can visually see how the content is laid out on the page.

For our wireframe design we created this design early on in the project and it served the purpose of showcasing the digital product we plan on making. Navbar is placed at the top with a carousel below it. The carousel will have at least two high quality images. The extinction main website has this feature so we wanted to follow it. Next comes the events divided into three sections, a feature that Bootstrap has thanks to it's grid design. Then we have a simple contact form followed by the footer at the end. Once we have the design completed. We had to change our mindsets into becoming web developers or programmers. Essentially, we take that design and break it up into its components.

Overall, the development process of a complete website is not a complex process as long as we follow a sequence of criterias or guidelines to make the website be done effectively. This section below follows how we divided and categorized our work and the road map used to complete this IT project. It describes a list of criterias that were followed in order to finish the project effectively and efficiently, which in turn helps to maintain, modify and update the system in the future.

Navigation

The explosive growth of the internet is based on the ability in which a user can move from one page to another using hyperlinks. Therefore, the linking structure of any website is very important. As Lohse (2006) says, "No amount of 'sparkle' in the presentation of products can overcome a site design with poor navigation features."

Navigation around a website can be achieved in a number of ways through navigation bars, graphic buttons, images and even keyboard shortcuts. However, developers should make content understandable and easy to navigate. This includes not only

making the language clear and simple, but also providing clear cut mechanisms for navigating between pages. Therefore, designers should present information in small sections with informative headings to produce which makers an easy read. There is a “three click rule”, saying that any information should not be further than three clicks away for the users (Brannan, 2010).

Our client, Extinction Rebellion has put great emphasis on navigation with the desire of allowing a user to find what they are looking for within the shortest amount of clicks or pages. This was because of the demographic users for their website being in their early twenties who are constantly on the go. Our client has kindly provided us with a sample web design template using InVision as seen in figure 4, 5 & 6.

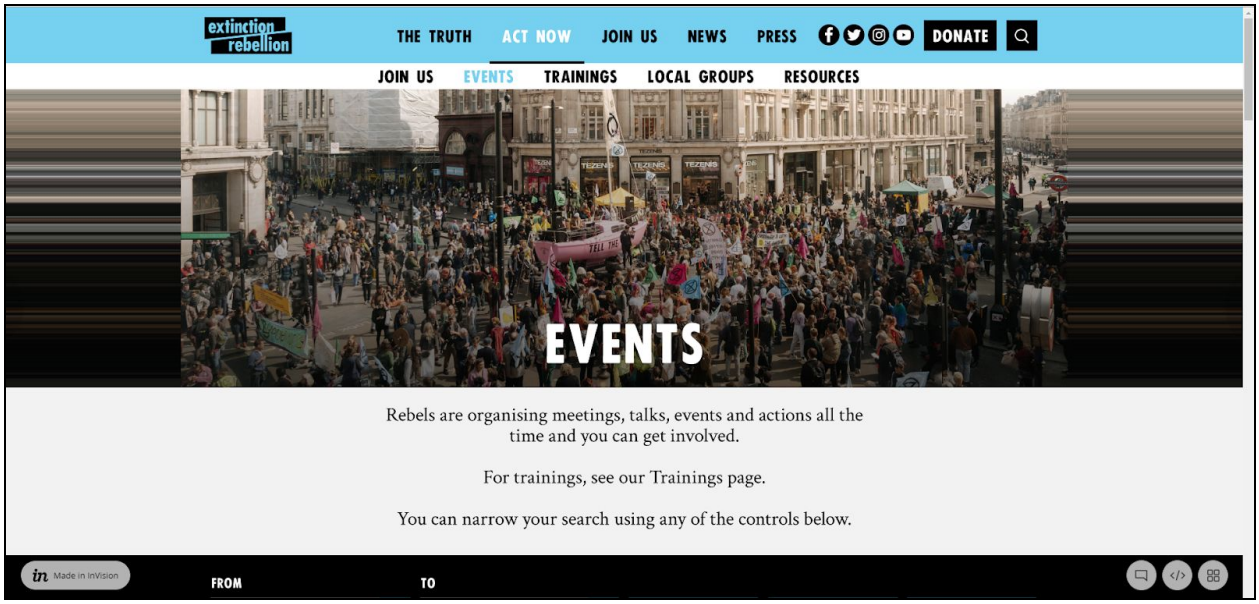

Figure 4: Web Design Template - Header.

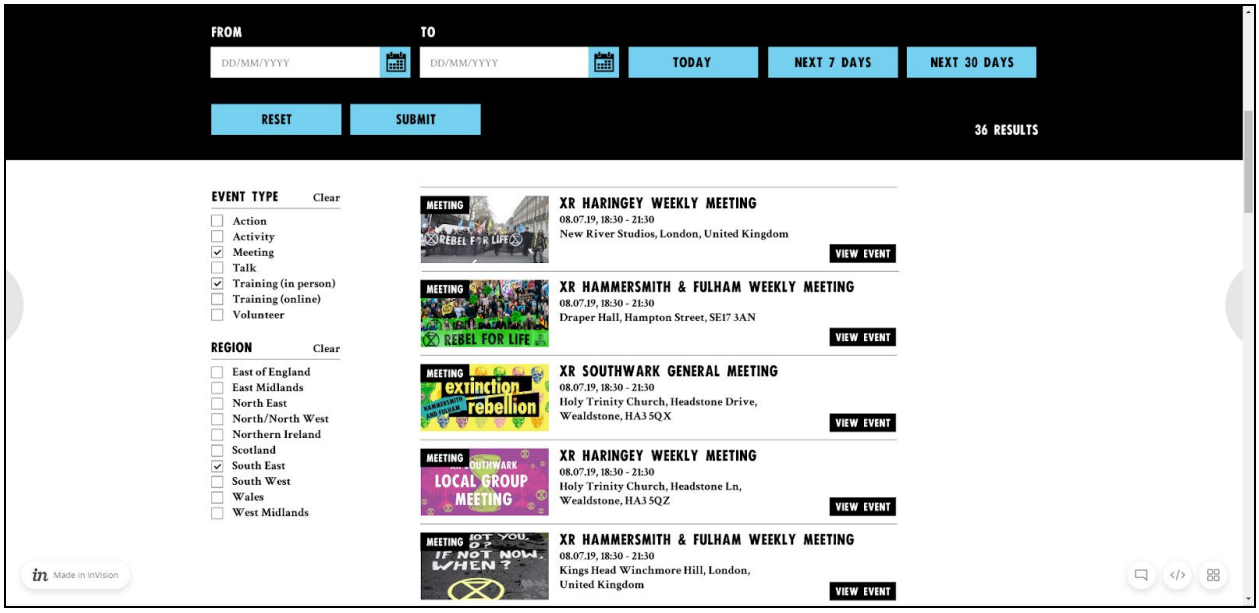

Figure 5: Web Design Template - Events.

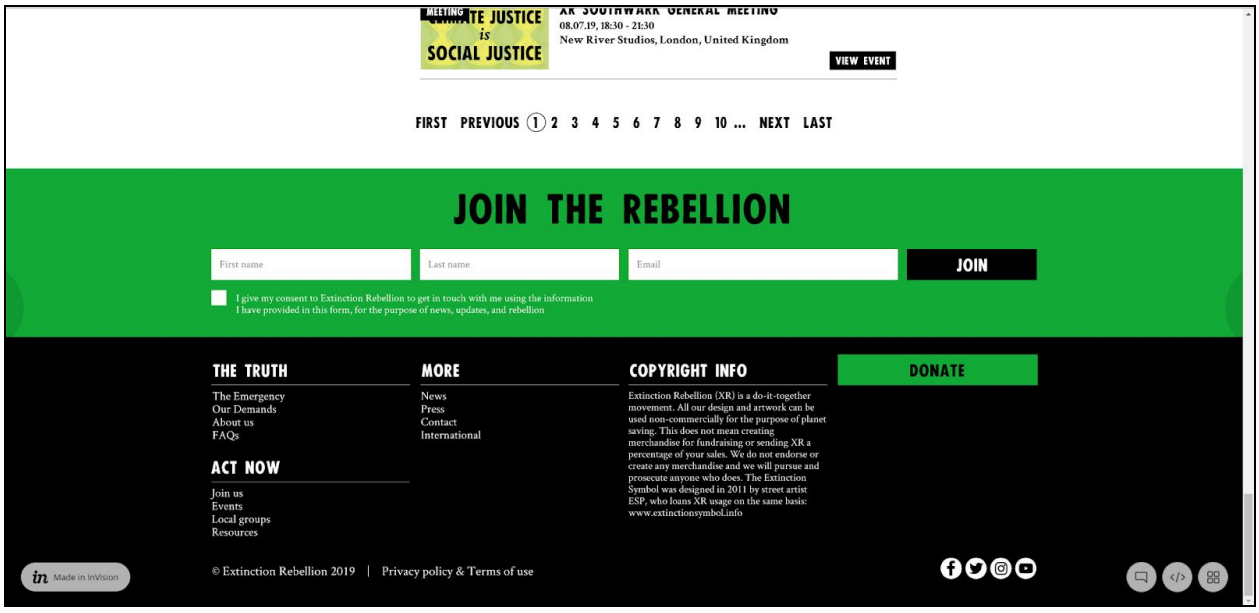

Figure 6: Web Design Template - Footer.

We mainly use it as a source of inspiration rather than a strict guide as our client provided no specific instructions to follow it word for word. This gives us a bit of leeway to implement personal adjustment that we find may be beneficial for our website. https://extinction-rebellion-project.herokuapp.com/

The main example would be to minimise the amount of links clickable in the navigation bar. We opted for only three headings within the navigation bar with the purpose of condensing everything to only three web pages worth of content to satisfy the point made above by our client. The main navbar becomes a drop down menu on smaller screens to make it easier for people to see what page they are looking for on mobiles or tablets. All pages display the identical Navbar, Footer and URL for titles. The drop down menu was also aligned in the center to match the overall layout of the website.

According to the BBC Future Media Standards & Guidelines (2014), the usage of only a navbar for our links provides a consistent style of presentation on each page that allows users to find content easier. A link, or a hyperlink, is a connection between two pages on the internet. With a link you can refer people to a page, post, image or other object on the internet, (Hallebeek, 2019). Such a mechanism creates a set of paths a user may take through the website, hence improving their probability of reaching the information they want in the website.

Consistency is very important for us, therefore, we opted out of the traditional blue/ purple underlining links for the headings and added a hover color that contrasts between the main navbar (blue) and sub navbar (white). We thought it may be distracting to see an underline appear whenever you are trying to click a heading.

We made sure that the header of each page is specific to the page and informative but also concise, restricting to only one word and eliminating linking words or propositions like the, an, a etc. The links need to be easily recognised and already identified

separate from each other, (Watrall & Siarto, 2008). Ultimately, “HOME”, “EVENTS” & “CONTACT” in uppercase were chosen as our page titles. Target=”blank” was added at the end to make all external links open a new tab in an attempt to keep users on our website instead of sending them to social media platforms.

All website pages must have the same navigation structure.	✓
Max 3 clicks is needed to reach any pages within the website from home.	✓
Footer navigation must be consistent in size and width.	✓
Links to open in a new tab to avoid directing users out of the website.	✓
Link text must describe the destination of the link.	✓
Order menu bar labels by frequency use and importance.	✓
Always use an identical URL to refer to the same page.	✓

Table 2: Design Guideline - Navigation.

Usability

“Usability refers to ensuring that interactive products are easy to learn, effective to use, and enjoyable from the user’s perspective.” (Preece et al, 2015). Usability is a fundamental level of user experience. According to Hartson et al (2012), “User experience is the totality of the effect or effects felt by a user as a result of interaction with, and the usage context of, a system device, or product, including the influence of usability, usefulness, and emotional impact during interaction, and savoring the memory after interaction.” Simply said, user experience design is an umbrella term for any kind of activity that provides better experience for the user. Without usability it is difficult to create a functional user experience.

This is important because if a product has bad usability, users are not able to achieve the aims in an efficient, effective and satisfied manner. They will leave the website and start to seek an alternative variant. A product with bad usability leads to bad user experience. Based on what users need from a website, developers must put content into the site.

Most people want to quickly scan information and read only small sections. If the information we have is in long paragraphs, we should consider adding many headings or use lines to separate text so people can find information quickly. Therefore, a thick black line is placed in between events on our website to clearly divide the text. It allows users to read easier with clear headings that show the title (larger font), description, start time and locations of each event. As our website was made with Bootstrap's grid system, this ensures that the content is responsive across all screen sizes.

It is also good practice for the information (events) to be displayed 10 results per page as a default through pagination, (Beerworth, 2012). According to Esser (2017), pagination is the process of splitting the contents of a website, or a section of contents from a website, into discrete pages. The very design and concept of pagination is to order content or results in a logical and digestible fashion, such that we can identify the most relevant result as easily as possible. A pagination gem called “Pagy” was added to meet the industry standard. However, we opted to put only 8 results as we felt the scrolling became too long.

According to Nielsen (2008), he defines a site map as a special page intended to act as a website guide. The term "site map" here thus encompasses a wide array of features, appearances, and names, including "guide," "overview," "index," and "directory." Site maps are also helpful as their main benefit is to give users an overview of the site’s areas in a quick look by giving a recap on all the links available within a website.

We have created a quick sitemap as seen below in figure 7. There's only one level of clicks for users to navigate from "Home" that's located in Navbar and Footer.

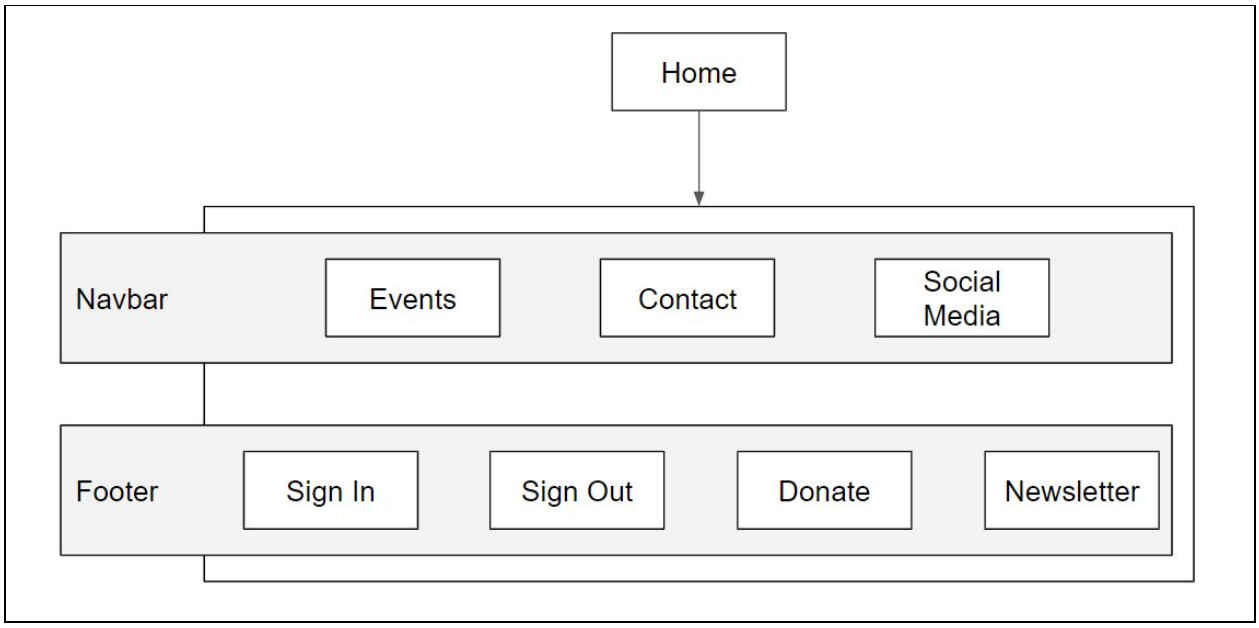

Figure 7: Sitemap Design

Users go to site maps when they are lost, frustrated or are looking for specific details on a crowded website. Since our website is minimal by design, we opted for a scroll to the top button and let our navigation bar act as a makeshift site map. This way, a user can simply click on the button wherever they are in the page and it will automatically scroll them back up to the navigation bar. However, a site map generator gem was also used but it's more for search engines rather than the website itself.

Lastly, a search bar was added to our website to further enhance the usability of the website. According to Babich (2017), search bar is a combination of input field and submit button. It allows people to search through a large collection of values by typing text into a field. The main criteria for a search function are that it provides accurate, short and to the point results and is easy to use.

A search bar allows users to quickly enter search terms and should allow the search to be further refined after they have seen the results. An advanced search allows users to perform the above feature and be able to define their initial search against specified search criterias or categories normally including searches on Metadata fields. To meet these requirements, a search bar and another option to filter by category was incorporated into the website.

Provide a site map.	✓
Every paragraph should be clear and concise.	✓
State the topic of the sentence or paragraph at the beginning.	✓
Consider providing quick search and advanced search facilities.	✓
Display search results in the most useful way.	✓
Design all pages to fit a mobile screen.	✓

Table 3: Design Guideline - Usability.

User Interface (UI)

User interface design is about the interaction between the user and the application or device, and in many cases, it is about the interaction between multiple users through that device, (Smashing Magazine, 2011). This means that user interface design isn't about how a product looks, but rather about how it works. It's not just about arranging buttons and picking colors, but rather about choosing the right tools for the job.

The guidelines here will discuss accessibility issues and provide design solutions for some problems that may face users. An example would be users not being able to see images due to a slow internet connection, turning off support for images to save data or

using text-based browsers that do not support images. The loading time of a website is one of the most critical aspects of building a responsive website. Paying attention to image size is critical for ensuring that responsive websites load quickly and deliver the best experience for mobile users.

Cao (2017) confirms that image size is a cause for the bad loading times in performances but also identified eight elements that can potentially cause performance issues in web design. These are bloated codes, plugins, themes, under optimized images, poorly built responsive design, render blocking Javascript & CSS, lack of GZIP compression and poor server setup.

Resolution in images refers to the amount of pixels per inch. Large image sizes mean unnecessary amounts of downloaded data for the user. Images can be resized with image editing software to minimise the amount of data (Niederst Robbins, 2007). Brannan (2010) also recommends using JPEGs for photographs and PNG files for everything else which we have followed for our project.

For this reason, a free online image compression tool (https://tinypng.com/) has also been used to make the image files smaller as we only used images that were taken from a professional camera. Poor quality icons and images gives the impression of low quality to the whole website. Compressing images greatly improves loading times through reducing the file size by selectively decreasing the number of colors in the image. The effect is nearly invisible but it makes a very large difference in file size. All our images also contain an “alt” text tag to provide a descriptive phrase about the image it accompanies.

According to Weinschenk (2011), “The primary purpose for using more than one font is to create an emphasis or to separate one part of the text from another. When many different fonts are used, the reader is unable to determine what is important and what is

not.” For the project, we limited ourselves to using three fonts, anything more and we thought the page can appear messy or unstructured.

The fonts were Franklin Gothic Medium, Crimson Text and Bootstrap's default font family “Helvetica Neue”. We chose Franklin Gothic because it was the closest match to the custom font used by Extinction Rebellion. We decided to avoid defining very specific fonts and opted out of using their custom font, as some users may not have them loaded on their machines. Crimson Text was chosen because we wanted there to be one least one serif font for the sake of contrast between headings and paragraphs. It is also a text that Extinction Rebellion has used before in their website.

Black font over a white background is the clearest among color combinations so no background images or colors were used for this project. The light blue heading and black footer provided a very nice contrast to the white background as it is. We also wanted to match our client’s template design.

As for the navbar and footer, we decided to go with light blue and black. Black and white are the easiest neutrals to add to just about any color scheme while adding a bright accent color to an otherwise neutral palette is one of the easiest color schemes to create (Willey & Sons 2011).

Avoid low quality graphics, animated gifs and blinking text..	✓
Contrast the foreground and background colors.	✓
Every image should be provided with an “alt” (alternative text attribute) tag.	✓
Use plain backgrounds to improve the readability of text.	✓
Use only a limited number of carefully chosen fonts.	✓

Table 4: Design Guideline - User Interface.

Accessibility

Web accessibility - the design and development of websites, tools and technologies so that people with disabilities can use them (Henry, 2019) - is essential in any context and particularly when considering the development of a digital tool for an organisation as diverse as XR. There are two key reasons why it is particularly important that we hold this focus during development: a) XR volunteers and interested parties represent a broad variety of different ages, backgrounds and abilities and b) one of the fundamental reasons our application is needed is to increase accessibility to information that has been previously difficult to collate and view in one place.

The major design function which represents accessibility in our application is the ability to tab through active links. We tried to ensure that everything on the homepage was accessible by keyboard and this access was tested by the client during a meeting towards the end of development (see Sprint 8 under Implementation). A key factor in ensuring this access was the decision not to use a date picker as part of the UI. It was very difficult to find a date picker which was fully accessible by keyboard. The initial plan was to use the ‘pickadate’ gem, however it was decided to structure the date-related menu options as drop-downs instead.

There was an additional request from the client regarding the ability for users to use the tab button to skip the navigational menu items by implementing a ‘skip to content’ function. Because this request was made relatively late in the design process, the full capability of the function was not able to be explored, however a function was implemented to tab to the filter menus by pressing ‘tab’ and then ‘enter’ on the load of the homepage. Additional work would be to explore skipping the RSS feed to jump straight from the event menus to the events themselves on the homepage.

Within the events section of the Homepage, a Facebook link has been added to each event to improve accessibility, This allows users to click on the event that interests them and enter Facebook immediately. The title of each event can also be used to access more information that was displayed as the show page.

Ensure keyboard accessibility for the website	✓
Provide a skip to content feature	✓
Every events should provide the relevant links	✓
Avoid keyboard inaccessible contents	✓

Table 5: Design Guideline - Accessibility.

Security

Information security, sometimes abbreviated to infosec, is a set of practices intended to keep data secure from unauthorized access or alterations, both when it is being stored and when it is being transmitted from one machine or physical location to another, (Fruhlinger, 2020).

There were several areas of concern within cyber security which directly affected the development of our application. There is the potential for OWASP standards to be compromised, including loss of sensitive data, potential for an injection attack and authentication errors. The Open Web Application Security Project (OWASP - https://owasp.org/) is an international organization dedicated to enhancing the security of web applications (IBM, 2015). There is also the risk of the application being targeted by hackers looking to specifically hinder or sabotage the work of XR for personal or political reasons.

An injection attack would mean that an attacker could trick our application into sending unintended commands through our search function. Our application (like most Ruby on Rails applications) relies on ActiveRecord, using methods on the models to query the data. As such, code within our application relating to search had to safely parameterise all queries. This meant ensuring that arguments were converted to arrays or hashes instead of strings. Passing user input directly to the query could result in users passing SQL queries directly into our application.

Within the application, we incorporated an admin control and login feature to the website through Devise gem. According to Schuessler (2018), creating a User that can log in and out of your application is so simple because Devise takes care of all the controllers necessary for user creation (users_controller) and for user sessions (users_sessions_controller).

Focusing on the design, this feature was added to allow admins to make changes to the events displayed. The login page was initially placed in the header but our client felt that it was too visible. As a result, we placed the admin control link to the footer in order to attract as little attention to it as possible.

Initially, we had two access levels, admin and user. Admin were able to access the event control page to refresh. add, edit and destroy events while users were only able to view the events detail with the show button. However, our client felt that the user access level was not a necessary requirement, therefore we took the ability to create user accounts away. This feature is also beneficial in avoiding any potential data leak or breach. However, for the purpose of this project, an user account is still available for testing to showcase the difference in access.

Data privacy or information privacy is a branch of data security concerned with the proper handling of data – consent, notice, and regulatory obligations. More specifically,

practical data privacy concerns often revolve around whether or how data is shared with third parties, how data is legally collected or stored and regulatory restrictions such as General Data Protection Regulation (GDPR - https://gdpr-info.eu/) or the California Consumer Privacy Act (CCPA - https://oag.ca.gov/privacy/ccpa) (Petters, 2020).

With the advancement of the digital age, personal information vulnerabilities have increased. Information privacy may be applied in numerous ways, including encryption, authentication and data masking - each attempting to ensure that information is available only to those with authorized access. These protective measures are geared toward preventing data mining and the unauthorized use of personal information, which are illegal in many parts of the world.

In order to avoid any potential issues with General Data Protection Regulation (GDPR), we avoided storing any information from users. A popular third party email service called Sengrid was used to handle all the data retrieved from users wishing to send emails via the contact form.

We made sure to not add any copyrighted images or texts in the website. This was possible through only adding information and assets that directly belonged to the Extinction Rebellion. To confirm, we had access and permission from the admins in Extinction Rebellion Facebook group to pull data from their events through the Facebook Graph API.

Due to the fact that secret API keys were used to connect to the Extinction Rebellion database, we decided to use a private Github repository to hide the API key. Hiding the API keys is necessary because it is an identifier for our access to our clients resource (Kcarrel, 2019). If the identifier is publicly available then someone else can use it to impersonate us and abuse our access to the resource which could lead to the whole project being shut down.

When framing the security concerns from a higher level, it is important to consider the risks inherent in developing a web application for an organisation which holds a strong political message at its core. Extinction Rebellion has even been described as having an ‘extremist ideology’ by terror police in the UK and their activists have been targeted by threats of violence as a result of their public activities. Cyber attacks against political organisations are common and, as well as this, any piece of software which brings more attention to the activities of a particular organisation runs the risk of inviting detractors or would-be attackers.

The Facebook developer account we were given access to is connected to three of the main XR Facebook groups and pages. We initially developed functionality to allow other XR Facebook administrators to authenticate through our application, allowing us to extract event data from their pages. However, there was a security risk in that an attacker could add inappropriate or false information to Facebook to defame XR which could then potentially be pulled through to the application. After discussion with Carl during sprint 8, we ascertained that this was not an essential requirement so we stopped development on this feature to prioritise other features.

Permission is obtained to use content sourced from other providers.	✓
Where appropriate, users are directed to other quality sites and sources that contain related information for Extinction Rebellion.	✓
Display copyrights statement when appropriate.	✓
When necessary, provide ID and password strategy.	✓
Provide a quick assistant if the user forgets ID or passwords.	✓

Table 6: Design Guideline - Security

Infrastructure

Even websites with high-end users need to consider download times: it has been found that many of users access websites from home computers in the evening because they are too busy to surf the web during working hours. Bandwidth is getting worse, not better, as the Internet adds users faster than the infrastructure can keep up.Due to the current pandemic, this trend has continued.

With the continous growth for web based applications, people have gotten more impatient and used to pages loading at very high speeds. Sadly, users don't care why response times are slow. In their mind, all they know is that the site doesn't offer good services. This is where slow response times often translate directly into a reduced level of trust and they always cause a loss of traffic as users take their business elsewhere.

Thanks to this trend, it has become a necessity to invest in a fast server, reviewing system architecture and code quality to optimize response times. We ultimately decided to use Heroku to handle our infrastructure. An in depth reasoning as to why we chose them can be found later in the dissertation as we continue to focus on web design in this infrastructure section.

There are other ways to ensure that users can read the content of our website with limited bandwidth that is more within our control. The most direct way would be to create an RSS feed. RSS is an acronym for Really Simple Syndication and Rich Site Summary. Essentially, RSS is an XML-based format created with the specific purpose of delivering updates to web-based content. Using this standard, webmasters provide headlines and fresh content in a succinct manner. Meanwhile, consumers use RSS readers and news aggregators (e.g. Feedly) to collect and monitor their favorite feeds in one centralized program or location.

According to Teske (2020), Netscape created RDF Site Summary which was the first version of RSS in March 1999. The slow rise to popularity is simple. RSS is free and an easy way to promote a site and its content without the need to advertise or create complicated content sharing partnerships. We decided to create and add a social media icon for our RSS feed in the navbar due to a number of advantages to RSS.

First and foremost, with RSS it is possible for users to see what's new in our website without opening a browser and enduring potential long loading times. In addition, the "feed checks" that deliver new content by the RSS reader are automatic, saving the tiring keystrokes and mouse clicks. Lastly, nothing is ever sent to an user email address. This is particularly useful to us as we have opted out of email gathering in order to erase the possibility of having hackers retrieve our users' sensitive data. A user can easily delete our feed through their RRS feeder. No messy email process is needed, just a click of the mouse or keyboard, and the RSS feed is gone.

Invest in good infrastructures	✓
Be aware of the caching time to be appropriate.	✓
Create RSS Feed.	✓

Table 7: Design Guideline - Infrastructure

Implementation

Sprint One - Finish 04/02/2020

The goal of Sprint 1 is to get our project running from the ground as soon as possible. For this purpose, we set up a shared Google folder and an IT tracker document which functions like a diary to keep a record of our tasks and ideas. We ultimately decided to use Trello as our main Agile board due to the reasons below:

- Rich features: You can create a number of boards within your personal or team workspace in Trello. These boards contain lists, which are groups of tasks. You can fill up the lists with individual tasks, which you can input in the form of cards with features like checklists and labels. .
- Easy to use: It works like a traditional whiteboard but in a digital form, the whole process of making cards and boards is exceptionally intuitive and user-friendly.
- Responsive design: Trello can be accessed or downloaded on the laptop, computer, smartphone, or tablet without any issue.
- Kanban system: A lot of other software uses the Kanban system which was developed by Toyota in the 1950s. Because of this system, it is now much easier for users to create tasks for projects and break these down into even smaller tasks by using its boards.

Next, we started looking for potential clients to fulfill the IT project requirement. Sarah ultimately connected with a developer that ended up being our client who functions as a middle person to the organisation called Extinction Rebellion. We had a few Skype meetings with Carl to discuss in detail the requirements for the project which can be seen above.

In an attempt to work as a team with our client (Carl), he had also kindly agreed to have monthly follow up Skype meetings with us to track progress, provide updates and to ensure that we were meeting the project requirements.

Next was to configure our operating system to use Rubymine. Unfortunately, Kit had issues when trying to implement this. This was because she started the project on Mac before trying to migrate it to her desktop. There were incompatibility issues as different ruby versions were used, resulting in conflicting gems. Since Sarah was also using windows, we ultimately decided to use Windows instead. Now it was just a matter of installing Rails and Git onto the system. Rubymine ensured that we were working on the same platform to minimise potential conflicts in the future when we try to merge branches in Github.

Sprint Two - Finish 28/02/2020

The main goal of Sprint 2 was to create a minimal viable product and get the application running. We first followed the tutorial on Rubymine's documentation website to create our Rails application. After that, we added Bootstrap and Jquery into the HTML file. As a private repository was already created, we connected the project to it using "git remote add origin". Fortunately, there were no major issues arising here as it was too still too early on in the project.

In terms of the Bootstrap layout, we started with creating a function navbar and made the necessary adjustments in route.rb. For now, an icon was added to the right and on the left hand side were the links. Below the navbar, a carousel was placed. The carousel is a slideshow for cycling through a series of content, prebuilt with CSS 3D transforms and a bit of JavaScript.

A placeholder for all the events were then placed along with a Google map and pagination at the bottom. It uses a large block of connected links for our pagination, making links hard to miss and easily scalable—all while providing large hit areas. Pagination is built with list HTML elements so screen readers can announce the number of available links. As we are trying to pull data from as many of the Facebook groups, it would make the application easier to use and improve readability. Lastly, a footer was placed to complete the layout.

As mentioned, Heroku was our chosen cloud platform as it lets companies or developers build, deliver, monitor and scale apps. This was where our first major issue for this project happened, when we found out that Sqlite3 doesn't work with Heroku becauses it uses a different type of database.

Turns out, each web application's database runs on a separate server (the database server). This means that our data won't appear on the Heroku server after deployment because Heroku will be pulling data from the database server instead of from your development.sqlite3 file. To overcome this issue, we had to install some extra gems to enable our web application to connect to the database.

Over the last few years, two open source database servers have become dominant on the Web. These servers are MySQL and PostgreSQL. In order to provide the best service available, Heroku decided to use Postgres as the default choice for all database hosting.

Another advantage is that Postgres is an open system and always will be (unlike MySQL). This means that as long as we are using PostgreSQL for our database server, we will not be subject to any vendor lock-in.

Naturally, we followed and installed Postgres. We used the tutorial provided by Heroku's devcenter to set up and got it to work at the start with relatively little difficulties. It wasn't until after pushing it to development that we realised the backend of the application had major incompatibility issues that will be explained in Sprint 3.

Sprint Three - Finish 13/03/2020

A major part of this sprint was finalising the means by which to pull event data from Facebook. The 'koalas' gem provided the means to return data from all Facebook pages associated with the developer account which had been created by Extinction Rebellion. We used this to return event data, defined a suitable schema in the database to capture event name, start and finish times and location and then populated the database.

Unfortunately, we then hit a setback in attempting to pass in Facebook group and page IDs in order to retrieve the rest of the Extinction Rebellion events. Without an association to the pages from the Facebook developer account, the data cannot be returned in this way. We therefore needed to use two additional gems - 'devise' and 'omniauth-facebook'- to add additional functionality to the application; the means for XR Facebook group and page administrators to give the application permission to pull event data. After speaking with Carl on the 16th March, we ascertained that the authorisation function would be optional - focusing on the events from the existing connection to the Facebook pages is fine for delivery of the application. There is a security concern, however, that we would need to ensure that only legitimate XR pages are able to be authorised through the application.

Kit also had an issue whereby she could not connect to development in Rubymines because of a "fe_sendauth: no password supplied" error. Therefore, she was unable to see any changes made within the layout of the application.

Numerous attempts were made to correct this error to no avail. Kit first started going to Stack Overflow and followed the instructions like changing the database.yml and gemfile to sort any potential routing errors but it didn't work. Then she moved on to re-installing Rubymine and Postgres with a new admin hoping this was the no password supplied error mentioned above which ultimately failed as well.

With no possible solution in sight, she changed the database back to Sqlite3 in order to see the development changes in the layout and when it was time to push to Github, the database.yml would be changed back to Postgres. This was in no way viable long term but acted as a short term fix as our schedules were being affected by this delay.

After Sarah successfully pulled data from Facebook, she merged the master branch to her personal branch. When Kit pulled the new master to her IDE, everything was magically working again. We are unsure as to what the issue really was but we hypothesized that during the Heroku migration from Sqlite to Postgres, there were some porting issues. Lastly, Kit reinstalled the Postgres a final time to match the user and password of the new database.yml.

We moved on to the next task, creating a search bar with a form helper. Forms use a POST or PUT/PATCH request. Search forms typically use GET requests because the data typically isn't confidential. GET requests display parameters in the URL and that usually isn't desirable for most forms (i.e. if the form has a password field, that data would get exposed to the public). This is to also facilitate the occasional user who may wish to bookmark a page of search results to refer back to it. We started with defining the method as GET and the action as our products index because we want our search results to load the Index page with the filtered listing. We then captured our search term and used it to filter our products.

Our client wants the ability to create accounts with admin controls so we decided to use Devise, a popular gem that provides all these user authentication functions. Devise can do many things but at its core it provides you with the ability to authenticate a user in your app using the register, sign in, and sign out functions. When you install the gem, the default settings will enable these basic functions but you can also customize the gem in many ways to add features for almost any possibility (reset password etc.).

Sprint Four - Finish 27/03/2020

This sprint contained work specifically to do with representing and filtering events within the application. This required us to ensure that events could easily be synchronised from Facebook and displayed appropriately to users. This was a complicated process which involved heavy configuration of how our application was communicating with the Facebook API.

A gem called “CanCanCan” has been added for authorization. Configuration came back with errors and was moved to Sprint 5.

Sprint Five - Finish 10/04/2020

This sprint marked the completion of some work not completed in Sprint Four, including the successful pull of data from Facebook and the ability for users to refresh event information within the application. Sarah took on the work of ensuring events were displaying in the application as Kit focused on successful deployment of the application and the front-end design.

Sprint Six - Finish 24/04/2020

Work continued on the implementation of the sort and filter function to allow users to manipulate the display of events they can see on the homepage of the application. We also met with Carl during this time to discuss any changes which may be necessary for the application, given the impact that Covid-19 is having on XR events. As XR's events have mostly shifted online, there is thankfully little need to adapt what we are doing. Facebook is still being used heavily to disseminate information about events.

Sprint Seven - Finish 08/05/2020

After spending 2 sprints trying to get Cancancan to work, we have ultimately opted for a more simplistic but hands on approach through Rails Console. It would have been a good feature to have for the client but due to the deadline creeping up, Cancancan was ultimately abandoned for this project.Emails can be granted with admin control by inputting the following code in the terminal:

- $ heroku run rails console
- @user = User.first
- @user.admin = true
- @user.save

Now we have successfully created 2 roles, user and admin. To access the events page, login is needed. Only the admin can see the refresh, edit, new and destroy event button. As we can no longer use "load_and_authorize_resource" from Cancancan, we Devist before: authenticate user and <% is user is signed in> instead to create the fields to be seen by admins. For the purpose of testing the application the login details are as below:

Username: admin@admin.com Password: password	Username: test@test.com Password: test1234

We also finished implementing the functionality to sort and filter events by category and date, after verifying that the application was correctly pulling data from Facebook. We had a catch-up meeting with Carl Hughes during this time and he was able to provide feedback on the current iteration of the application. The summarised feedback included some changes which were not initially discussed within the user requirements:

- We should minimise any reference to logging in to the application or accessing any administrative functionalities. Login information can be kept as information only inside XR
- We should change the images being pulled from Facebook so that they are representing the group rather than the individual event
- We should ensure that event categories can be set by administrators. Any logic to set them within the application is acceptable, however they should be able to be changed.

Sprint Eight - Finish 22/05/2020

Unfortunately, we ran into more issues as our website on Heroku was broken. Following the websites advice, we ran the application log --tail and noticed the error:

Heroku[router]: at=error code=H10 desc="App crashed" method=GET path="/favicon.ico" ...

At first, we thought it could be an issue with the asset but after trying heroku run rails db:migrate, we found out there was a migration issue in the PG table. With the deadline so close, we asked our lecturer, Bruce for assistance.

He pointed us to the right direction and concluded that the issue was specifically with db/migrate whereby a migration file was missing. We checkout the branch where the missing file was located and merge the master file. After this, we were able to successfully db:migrate to Heroku. Unfortunately, the error still exists but at least now, we know the database is working again.

According to Eagles (2020), Heroku H10-App Crashed Error were usually caused by four reasons: bug in Procfile, setting a PORT as a Heroku environment variable, missing required environment variable and missing required scripts. After the process of eliminations, we concluded it was missing the npm file thanks to the error: npm err! missing script: start. All we needed to do was run npm install and Heroku was finally working again.

An RSS feed was one of the last features we discussed with our clients. With the abundance of resources in RSS creation and lack of time, we chose to use a third party website to fulfil to create and display an RSS feed as a side widget for our website. There were two parts for this feature, the first is to pull and display the RSS feed for https://rebellion.earth/news/ and feed.mikle.com was used to complete this feature. It's a widget creator that allows you to display the RSS of your chosen xml file. Minor styling options are available and we made the width 300px and put it in the sidebar below the filter search feature.

The second part is to create a RSS feed for our project. This was done through a handy website called http://fetchrss.com/. We simply entered our website address and chose the area we want the RSS feed to display which includes the title, description, time, location and facebook link. The RSS feed was displayed along with the social media icons in the navbar and the footer section of the website.

The facebook link was added after we spoke to our lecturer, Bruce. It was a simple add-on that we ultimately overlooked during the process of creating the admin control. For this add-on, we inserted <a href="https://www.facebook.com/events/<%= *event*.fbid %>" target="_blank">https://www.facebook.com/events/<%= *event*.fbid %>.

During this sprint, we also implemented the logic for allowing administrators to update event information without it being overwritten by the Facebook synchronisation (see page 45).

Sprint Nine and final stage of development - Finish 05/06/2020

We met with Carl again on the 26th May to discuss final additions or changes needed to the application. These included:

- The logos should be inline with the banner menu items (✓)
- Borders should be added to stop the RSS feed from running into the events (✓)
- Mobile responsiveness needs to be checked (✓)
- Ideally the content should match the width of the header image. (✓)

This involved some accessibility testing where he was able to check that the application's main functionalities were accessible through tabbing. Choosing not to use a date picker has helped this to be possible and fulfil this requirement. An accessibility feature allowing users to skip straight to the content using the tab button has been requested. We will see if we can implement this in the time remaining. In general, we have delivered the agreed requirements and the client is happy to continue development work on the application with XR.

Several minor changes have been implemented during this time, including additions to the infoboxes on the Google map markers and ensuring that all URLs are clickable

when they appear within the text. There was also the addition of a function to refresh a single event from Facebook, rather than needing to sync all the events at once. This was not a client request but a way to ensure we could test the API during development without hitting any limits.

To meet the requirements set by Carl, numerous last minute changes were made to the front end design of the website. In regards to the logo, we changed it from .png file to .svg as the image wasn't sharp enough when it was displayed on HD screens. This feature was completed by using the HTML code for the svg to the Navbar section of the application.html.erb. The logo was set to: width="150px" height="50px" to ensure it's inline with the other menu items in the banner. A border was also added to the sidebar with border-style: solid.

Lastly, we checked the CSS file and realised that the assets weren't pushing to Heroku. This was the reason why it's not displaying any of the changes made in any of the assets such as the correct grid system and margins for responsive layout. To solve this, we had to add input the following in our terminal for Window:

- $ set RAILS_ENV=production
- $ bundle exec rake assets:precompile
- $ git add public/assets -f
- $ git commit -m "vendor compiled assets"

It is very time consuming and a hassle but this issue only came up within the last 2 weeks of the project deadline so we opted to leave it as it is, instead of finding a long term solution.

During this sprint, we have also been able to implement the help guide for the application and responded to feedback from our presentation on the 3rd June as follows:

- **Security** - We reviewed the code which allows the users to input information (such as the ‘index’ method for search in the Event controller) to ensure that params were used instead of strings. This helps us protect against SQL injection attacks. We also used an environment variable in Heroku to secure the Facebook API key rather than include this in the code.
- **Accessibility** - We implemented a ‘tab to content’ function allowing users to skip from the nav bar to the event filter menus on the homepage. This was originally requested as an extra feature by the client.
- **Testing** - We implemented additional tests for the Event and Eventedit models to check for the presence of attributable data.
- **Transferring code to the model** - We refactored the code so that the ‘synch events’ method is contained in the Event model rather than the controller.

Reflecting on the project

Developing any software application as a team requires numerous skills for success. Although programming skills and an understanding of software architecture are both important, we have also needed to utilise skills in communication, business analysis and project management throughout development. In addition, both developers have had to balance the demands of the project with full-time work and all of the changes to daily life made inevitable by the Covid-19 crisis.

In the initial stages of the project, it was important that we choose a client and project quickly so as to maximise development time. This was largely due to the unpredictability of a) developing a project completely remotely and b) both developers needing extra time to learn the tools and fix any bugs during development. Two key choices helped this process: a) choosing to work with an individual who was already known to one of the developers and b) choosing a project which had measurable aims and goals and where features were unlikely to spiral out of control.

Although balancing the demands of gathering requirements, designing the application, dealing with data and writing tests were tricky to handle between a team of two, we were lucky that our interests and skills were at least partially in alignment. Sarah took the lead on most of the backend work (including connecting to the API, storing data and developing logic for updating data) and Kit took on the majority of frontend development and the design of the application.

Despite choosing an application which we felt would be manageable to develop, we still needed to ensure that we had a robust set of requirements to satisfy both the client and the requisites for the project. In order to ensure that we delivered these requirements, we tracked them closely throughout development, checking progress with every meeting and using a Trello board to sort and prioritise tasks. We also made sure to

check the requirements at every meeting with Carl Hughes to ensure that we were not prioritising incorrect requirements or spending too long on features that had not been requested. A Google document was used to keep short minutes from each catch-up which contained more detail than could be contained on the Trello board.

By keeping development so tightly focused on the requirements, we were able to ascertain which features were dealbreakers for the client and which were not. For example, being able to edit event information was crucial to the application's success. However, it was more difficult than originally envisaged to allow administrators to authenticate their individual Facebook pages through the application and allow us to pull more event data. After speaking with Carl, we could find out that this was not absolutely necessary for completion of the application and so decided to stop working on this code to prioritise other features.

One of our main challenges during development was not having a clear picture of how long features would take to be developed or how much effort would be required. There may not have been a way to avoid this challenge (as this is a reasonable expectation for a student project) however, we were able to mitigate the risk involved by scheduling regular meetings on Skype to discuss progress. These took place once a week for the majority of development and then twice a week on the run-up to the deadline. Having a space to discuss successes and problems helped us both to learn from each other, acknowledge the steep learning curve for our development and persevere when setbacks occurred. We also communicated over Whatsapp in between meetings if there were any major issues or to alert each other to big changes to the application.

In addition, it was very useful for the progress of development to use the Agile methodology of using 'sprints' to break tasks up into two-week chunks. This way we could review the pace of work, more accurately estimate when we needed to stop

working on something and have a deadline for tasks so that they did not run on unnecessarily long. It also helped us to document our progress accurately.

In the context of our circumstances, we have worked hard to stay on track, ensure that we completed the requirements and coped with the myriad of changes made to our daily routines by the current global crisis. Any discussion of how we would do the project differently needs to be considered in this context. Based on our learning during this project, some of the recommendations for changes could be:

- Creating a structure for the data schema before development. We implemented several tables where data was held so that it could be effectively edited in the application and displayed. However, with a larger data structure involving more tables we may have had more flexibility about how to adapt and change the structure of the application.
- Involving an additional representative from the client to add another voice in the stages of development and add more detail to requirements and more rounded feedback.
- Considering using an alternative tool to Heroku for application deployment, such as Google Cloud.

References

Agilemanifesto.org, (2001). *Agile Manifesto official website.* [online] Available at: http://agilemanifesto.org/ [Accessed 03 Mar 2020]

Almeida, F., & Monteiro, J. (2017). *Approaches and Principles for UX web experiences.* International Journal of Information Technology and Web Engineering, 12(2), 49-64.

Altexsoft, (2018). *Functional and Nonfunctional Requirements: Specification and Types*. [online] Available at: https://www.altexsoft.com/blog/business/functional-and-non-functional-requirements-specification-and-types/ [Accessed 20 Mar 2020]

Ambler, S., (2004). *The Object Primer: Agile Model-Driven Development with UML 2.0 (3rd Ed).* New York: Cambridge University Press. pp134-153

Arnuphaptrairong, T. (2011). Top Ten Lists of Software Project Risks: Evidence from the Literature Survey. In: *Proceedings of the International MultiConference of Engineers and Computer Scientists.* [online] Hong Kong: IMECS II. Available at http://www.iaeng.org/publication/IMECS2011/IMECS2011_pp732-737.pdf [Accessed 16 Apr 2020]

Barbich, N (2017). *Design a Perfect Search Box.* UX Planet. [online] Available at: https://uxplanet.org/design-a-perfect-search-box-b6baaf9599c [Accessed 26 May 2020]

BBC Future Media Standards & Guidelines (2014), *Text Links Standards v2.1.* [online] Available at: http://www.bbc.co.uk/guidelines/futuremedia/accessibility/links.shtml [Accessed 26 May 2020]

Beerworth, R. (2012). *Take advantage of pagination in your website design.* [online] Available at: https://www.wiliam.com.au/wiliam-blog/web-design-sydney-take-advantage-of-pagination-in-your-website-design [Accessed 26 May 2020]

Brannan, J. A. (2010). *Web Design.* Edinburgh: Pearson Education Limited, pp8, 39, 64.

Brizard, T., (2015). *Broken Agile*. New York: Springer Science & Business Media, pp33-36

Cao, J. (2017). *Important considerations for responsive design performance & UX.* Available at https://www.uxpin.com/studio/blog/important-considerations-responsive-design-performance-ux/ [Accessed 22 May 2020]

Castello, J., (2018). *The Definitive Rspec Tutorial With Examples*. [online] Ruby Guides. Available at https://www.rubyguides.com/2018/07/rspec-tutorial/. [Accessed 14 Mar 2020]

Dasso, A., & Funes. A., (2007). *Verification, Validation and Testing in Software Engineering.* London: Idea Group Publishing, pp4

Douglass, B., (2016). *Agile Systems Engineering*. Waltham: Morgan Kaufman, pp41-84

Eagles, L. (2020). *Causes of Heroku H10-App Crashed Error And How To Solve Them.* Dev.to.[online] Available at: https://dev.to/lawrence_eagles/causes-of-heroku-h10-app-crashed-error-and-how-to-solve-them-3jnl [Accessed 06 Jun 2020]

Edwards, S. (2003). *Using Test-Driven Development in the Classroom: Providing Students with Automatic, Concrete Feedback on Performance.* [online]. Virginia Tech University. Available at https://www.cs.tufts.edu/~nr/cs257/archive/stephen-edwards/automated-feedback.pdf [Accessed 06 Mar 2020]

Esser, P. (2017). *Split the Contents of a Website with the Pagination Design Pattern.* [online] Available at: https://www.interaction-design.org/literature/article/split-the-contents-of-a-website-with-the-pagination-design-pattern [Accessed 26 May 2020]

Extinction Rebellion website. (2020). *About us* [online] Available at: https://rebellion.earth/the-truth/about-us/. [Accessed 04 Jun 2020]

Extinction Rebellion website. (2020). *History* [online] Available at: https://rebellion.earth/2020/03/02/extinction-rebellion-history-2018-2019/. [Accessed 15 Jun 2020]

Extinction Rebellion website. (2020). *Smartphone: Friend or foe?* [online]. Available at: https://rebellion.earth/2020/02/01/smartphone-friend-or-foe/. [Accessed 15 Apr 2020]

Extinction Rebellion website. (2020). *Tech Action & Solidarity Meeting* [online]. Available at: https://www.xrebellion.nyc/events/heading-for-extinction-and-what-to-do-about-it-8619-darwz-wm6aw-kjakr-e9k6j-fwsmx-rjrlc-pzfa5-x9tfy-bkhhr-ef6bf-9amta-a4wy9-gn7dc. [Accessed 18 Jun 2020]

Extinction Rebellion website. (2020). *Volunteer roles*. [online]. Available at: https://volunteer.extinctionrebellion.uk/roles [Accessed 04 Jun 2020]

Fruhlinger, J. (2020) *What is information security? Definition, principles, and jobs.* [online] Available at: https://www.csoonline.com/article/3513899/what-is-information-security-definition-principles-and-jobs.html [Accessed 26 May 2020]

Gamble, A., Carneiro Jr., C., Al Barazi, R. (2013) *Beginning Rails 4*. Berkeley: Apress, pp1-11

Gathering Requirements: The agile way. (2020). [video] University of Minnesota. [online] Available at https://www.coursera.org/lecture/agile-software-development/gathering-requirements-the-agile-way-FdJzE [Accessed 16 Apr 2020]

Gibson, J.E., Scherer, W.T., Gibson, W.F., Smith, M.C. (2016). *How to Do Systems Analysis: Primer and Casebook.* Hoboken: John Wiley & Sons, pp.4-5

Gilb, K., (2007). *Evolutionary Project Management & Product Development.* [online] Available at https://www.gilb.com [Accessed 25 Mar. 2020]

Hallebeek, W. (2019). *SEO Basics: What is link building?* [online] Available at: https://yoast.com/what-is-link-building/ [Accessed 26 May 2020]

Hartl, M., (2020). *Ruby on Rails Tutorial*. Massachussetts: MIT [online] Available at: https://www.railstutorial.org/ [Accessed 10 Mar 2020]

Hartson, Rex & Pyla, Pardha (2012). *The UX Book: Process and Guidelines for Ensuring a Quality User Experience.* Waltham: Elsevier.

IBM, (2014). *OWASP top 10 vulnerabilities.* [online] Available at: https://www.ibm.com/developerworks/library/se-owasptop10/ [Accessed 26 May 2020]

Kcarrel, (2019). *Major Key Alert: Hide your API Keys.* Devpost. [online] Available at: https://dev.to/kcarrel/major-key-alert-hide-your-api-keys-l4c [Accessed 26 May 2020]

Kiely, P. (2019). *Adapting Agile For Part-Time Teams*. [online]. Smashing Magazine. Available at Adapting Agile For Part-Time Teams — Smashing Magazine [Accessed 23 Mar 2020]

Laporte, C., & April, A., (2017). *Software Quality Assurance*. Hoboken: John Wiley & Sons, pp.89-90

Lemon, K. (2018) *How I learned to stop worrying and love TDD.* [online] Hackernoon. Available at https://hackernoon.com/how-i-learned-to-stop-worrying-and-love-tdd-ea22b7b7fcaa [Accessed 22 Apr 2020]

Lewis, C. and Rieman, J. (1994). *Task-centred User Interface Design*. Boulder: Lewis & Rieman. Pp.100-110

Lohse, G. (2006). *Internet retail store design: How the user interface influences traffic and sales.* Journal of Computer-Mediated Communication, 5 (2). Retrieved on May 22. https://onlinelibrary.wiley.com/doi/full/10.1111/j.1083-6101.1999.tb00339.x

Martin, A. (2018). *Ruby, Rails, and the 4 Principles of OOP*. [online] Medium. Available at: https://medium.com/@autumnmartin.dev/ruby-rails-and-the-4-principles-of-oop-7836a54074e6 [Accessed 25 May 2020]

Mayers, D. (2017). What is the difference between a back end and a front-end web programming language? [online] Kevin Chisholm - Blog. Available at: https://blog.kevinchisholm.com/web-development/difference-between-back-front-end-language/ [Accessed 25 May 2020]

Mears, C (2013). *The UX review, Wireframes – The Beginner's Guide. WWW publication.* Available at https://theuxreview.co.uk/wireframes-beginners-guide/ [Accessed 20 May 2020]

Measey, P., Wolf, L., Berridge, C., Gray, A., Levy, R., Oliver, L., Roberts, B., Short, M. and Wilmshurst D. (2015). *Agile foundations: Principles, practices and frameworks.* Swindon: BCS Learning and Development, pp17

Metz, S. (2016). *The Wrong Abstraction*. [online] Sandi Metz. Available at: https://www.sandimetz.com/blog/2016/1/20/the-wrong-abstraction [Accessed 25 May 2020]

Metz, S. (2019). *Practical Object-Oriented Design (2nd Ed)*. New York: Addison-Wesley. pp.11

Modelling and the UML. (2020) Open University. [online] Available at: https://www.open.edu/openlearn/science-maths-technology/introduction-software-development/content-section-6. [Accessed 6 May 2020]

Morville, P. and Rosenfeld, L. (2007) *Information Architecture for the World Wide Web*. Sebastopol: O'Reilly Media. pp.91-95

Mulonda, Y. (2018) *Object-Oriented Programming Concepts "In Simple English"*. [online] Noteworthy. Available at: https://blog.usejournal.com/object-oriented-programming-concepts-in-simple-english-3db22065d7d0 [Accessed 25 May 2020]

Nielsen, J. (2008). *Site Map Usability.* [online] Available at: https://www.nngroup.com/articles/site-map-usability/ [Accessed 26 May 2020]

Ogden, C. (2019). *Interview: Mike Berners-Lee on why there is no Planet B*. [online] Environment Journey. Available at: https://environmentjournal.online/articles/interview-mike-berners-lee-on-why-there-is-no-planet-b/ [Accessed 03 Apr. 2020]

Peters, C., (2012). *3 Key Software Principles You Must Understand*. [online] Envatotuts. Available at https://code.tutsplus.com/tutorials/3-key-software-principles-you-must-understand--net-25161 [Accessed 16 Apr 2020]

Patton, J., (2005) *It's all in how you slice: Designing your project in working layers to avoid half-baked incremental releases.* Better Software, (January 2005), pp.16-40

Patton, J. (2020) *The New User Story Backlog is a map*. [online] Jeff Patton & Associates. Available at: https://www.jpattonassociates.com/the-new-backlog/. [Accessed 02 May 2020]

Radigan, D. (2020). *Story points and estimation.* [online] Atlassian. Available at: https://www.atlassian.com/agile/project-management/estimation [Accessed 10 Apr. 2020]

Rails Guides. (2020). *Getting Started With Rails.* [online] Available at https://guides.rubyonrails.org/getting_started.html [Accessed 15 Apr. 2020]

Rails Guides. (2020). *Testing Rails Applications*. [online] Available at https://guides.rubyonrails.org/testing.html#rails-sets-up-for-testing-from-the-word-go [Accessed 12 Mar 2020]

Petters, J. (2020). *Data Privacy Guide: Definitions, Explanations and Legislation.* Varonis. [online] Available at: https://www.varonis.com/blog/data-privacy/ [Accessed 26 May 2020]

Niederst Robbinds, J. (2007). *Learning Web Design.* Sebastopol, California, USA: O'Reilly Media.

Orlando, D. (2011). *Cloud computing service models, Part 2. Platform as a Service.* IBM Developer. [online] Available at: https://www.ibm.com/developerworks/cloud/library/cl-cloudservices2paas/ [Accessed 27 May 2020]

Ruby for Beginners. (2020). *Object-oriented programming.* [online] Available at http://ruby-for-beginners.rubymonstas.org/object_oriented_programming.html [Accessed 5 Apr 2020]

Ruhe, G. and Wohlin C. (2014). *Software Project Management in a Changing World*. Heidelberg: Springer-Verlag, pp.104-120

Saffer, D., (2010). *Designing for Interaction, Second edition: Creating Innovative Applications and Devices*. Berkeley: New Riders, pp.44-45

Shneiderman, B., and Plaisant, C. (2010) *Designing the User Interface: Strategies for effective human-computer interaction (5th edn).* Harlow: Pearson Education Ltd. pp.88-95

Siau, K., Chiang, R. and Hardgrave, B.C. (2011). *Systems Analysis and Design: People, Processes, and Projects.* New York: Taylor & Francis Group, pp.4-10

Smashing Magazine, (2011). *User Interface Design in Modern Web Applications.* [online] Available at: https://www.smashingmagazine.com/user-interface-design-in-modern-web-applications/ [Accessed 26 May 2020]

Software Project Planning & Control, (2013). *University of Aberdeen course notes*

Subic, N., Krunic, T., & Gemovic, B. (2014). *Responsive web design – Are we ready for the new age?* Online Journal of Applied Knowledge Management, 2(1), 93-103.

Schuessler, M. (2018). *Devise Authentication with Rails 5.* Gitconnect. [online] Available at: https://levelup.gitconnected.com/devise-authentication-with-rails-5-815b5a9d6daf [Accessed 26 May 2020]

Teske, C. (2020). *What Is an RSS Feed? (And Where to Get It).* Lifewire. [online] Available at: https://www.lifewire.com/what-is-an-rss-feed-4684568 [Accessed 06 Jun 2020]

Vanderjack, B., (2015). *The Agile Edge: Managing Projects Effectively Using Agile Scrum.* New York: Business Expert Press, pp.15-20.

Wang, Y., (2017) *SQLite vs. MySQL vs PostgreSQL: A Comparison Of Relational Database Management Systems*. [online] Medium. Available at: https://medium.com/@yangforbig/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems-afd5afd6566 [Accessed 16 Jun 2020]

Watrall, E. & Siarto, J. (2008). *Head First Web Design.* Sebastopol, California, USA: O'Reilly Media, pp29, 208.

Weinschenk, S (2011). *100 Things Every Designer Needs to Know About People (Voices That Matter)*. San Francisco: New Riders.

Willey, J & Sons (eds.) 2011. Smashing Magazine. Professional Web Design. United Kingdom: John Wiley & Sons Ltd, pp100.

Williams, R. (1995). *Self-Directed Work Teams: A Competitive Advantage*. [online] Quality Digest. Available at https://medium.com/commencis/benefits-and-challenges-of-self-directed-teams-in-software-projects-fe8df2d842e8 [Accessed 03 May 2020]

Appendix

XR Centralised Events Application

Developers: Kit Man Liew and Sarah Rudston
Client representative: Carl Hughes

Goals

Our goal is to create a web application which will provide a centralised place for Extinction Rebellion volunteers and supporters to find information about events, both national and local.

Background & strategic fit

Extinction Rebellion is an international movement that uses non-violent civil disobedience in an attempt to halt mass extinction and minimise the risk of social collapse.

Anyone can be a part of Extinction Rebellion by joining or starting a local group in their area. These local groups organise their own events and meetings and there are also large-scale events promoted nationwide.

Because there are many small Extinction Rebellion groups, there are a lot of places online where information can be held about events. This makes it difficult for volunteers to easily find a full picture of what is going on and to receive event updates efficiently. The organising groups often post information on Facebook which is difficult for people who do not have Facebook accounts. It is also difficult to maintain consistency about how events are publicised.

User stories

User story	Description	Priority	Notes
Event display	A user wants to see events displayed in a list	**Must have**	
Event sort and filter	A user wants to filter events by location, date and type of event	**Must have**	
Recurring event display	A user wants to see recurring events displayed in a logical way	**Must have**	
Location of events	A user wants to see local events displayed on a Google map	**Must have**	
Admin controls	An administrator wants to log in to the application to make manual updates to events	**Should have**	
RSS feed of events	A volunteer wants to use an RSS feed to pull information about events to their own website	**Should have**	
Replace images	An administrator wants to replace default images from Facebook	**Should have**	
Social media posts embedded	A user wants to see relevant social media posts embedded in the application, linked to events	**Nice-to-have**	

Accessibility	A user wants the application to be accessible	**Must have**	Ensuring the application is keyboard accessible and accessible to screen-readers
Security	A user wants the application to be secure	**Must have**	There are often attempts to take down XR online presences