The University of Sheffield

Faculty of Engineering

Department of Computer Science



A Time Traveller’s Map

Oliver Rahman

Supervised by Dr Siobhan North

COM3600

May 2018

Signed Declaration

All sentences or passages quoted in this report from other people’s work have been speciﬁcally acknowledged by clear cross-referencing to author, work and page(s). Any illustrations which are not the work of the author of this report have been used with the explicit permission of the originator and are speciﬁcally acknowledged. I understand that failure to do this amount to plagiarism and will be considered grounds for failure in this project and the degree examination as a whole.

Name: Oliver Rahman

Signature:

Date:

**Abstract**

Contents

[Chapter 1](#_Toc1557078)

[1.1 Project Description 6](#_Toc1557079)

[1.2 Project Justification 6](#_Toc1557080)

[1.3 General Analysis 6](#_Toc1557081)

[Chapter 2](#_Toc1557082)

[2.1 An Introduction to GIS 7](#_Toc1557083)

[2.2 Existing Mapping Systems 8](#_Toc1557084)

[2.1.1 Map Storage and Search 8](#_Toc1557085)

[2.1.2 Mapping Functions 9](#_Toc1557086)

[Chapter 3](#_Toc1557087)

[3.1 System Overview 11](#_Toc1557088)

[3.2 Requirements 12](#_Toc1557089)

[3.2.1 Requirements Overview 12](#_Toc1557090)

[3.3 System Architecture Decisions 13](#_Toc1557091)

[3.3.1 Platform 13](#_Toc1557092)

[3.3.2 Programming Language 14](#_Toc1557093)

[3.3.3 Tools and Libraries 14](#_Toc1557094)

[3.3.4 Framework 14](#_Toc1557095)

[3.3.5 Front End 14](#_Toc1557096)

[3.3.6 Database 15](#_Toc1557097)

[3.3.7 DBMS and Mapping Data 16](#_Toc1557098)

[3.4 Anticipated Development Problems 16](#_Toc1557099)

[3.5 Ethical Considerations and Pre-Implementation Changes 17](#_Toc1557100)

[3.6 Specific Requirements 18](#_Toc1557101)

[Chapter 4](#_Toc1557102)

[4.1 User Interface Design 20](#_Toc1557103)

[4.1.1 General Layout 20](#_Toc1557104)

[4.1.2 World Events 21](#_Toc1557105)

[4.1.3 Timeline 22](#_Toc1557106)

[4.1.5 Map Annotation 22](#_Toc1557107)

# Chapter 1

Introduction and Overview

## 1.1 Project Description

This project – A Time Traveller’s Map – is an endeavour to build a piece of useable, expansive and adaptive mapping software for a wide range of users, but primarily geared for educational use. The final product will display detailed map data, in chronological order, and designed in a way as to be easily navigable and customisable. Users should be able to share their historical data from any time period, at any level of granularity and with as much additional information as they wish to display, all while being pleasant to looks at and easy to use. Users will be able to share their maps with other users, who should be able to easily access these newly created timelines.

An example of the intended use of this software would be to link a series of maps showing the expansion and contraction of the Roman nation between the years 510BC and 1453AD. Users should be able to view individual turning points – The Punitive War, The Final War of the Roman Republic, etc - and should be able to ‘zoom in’ on individual events, seeing battle maps, troops deployments and other related information. This transition between time periods should be facilitated with a slider bar as well as any other suitable methods deemed contributory to ease of use during the Literature Survey.

The first two chapters of this report will analyse the existing marketplace for similar technology, examine and compile the best aspects of these systems, before breaking the potential application into its component requirements in preparation for design.

## 

## 1.2 Project Justification

## 1.3 General Analysis

- To be done on completion –

# Chapter 2

Literature Survey

This project will largely be a software development undertaking and so a significant part of the literature review will be examining existing software solutions. Investigating current applications will mostly involve compiling a list of features that work well, features that could be improved, and features that do not work well. As this analysis is completed, special attention will need to be paid towards finding a niche in functionality that the potential system can work within or a better way to gather and display information that results in a more user-friendly system.

Note that for ease of reading, the Literature Survey has been divided into two chapters, Chapter Two will primarily be an analysis of existing mapping solutions and approaches in order to get a wide understanding of current standards. Chapter Three will discuss appropriate software, development techniques and will provide explicit requirement definitions of the project.

## 2.1 An Introduction to GIS

The Esri institute defines GIS as “A framework for gathering, managing, and analysing data, rooted in the science of geography.” (Esri, 2017) GIS arose from the need to convert paper maps into a digital form for easy manipulation and access. “The sheer volume of information meant that areas that are large with respect to the map scale could only be represented by a number of map sheets” (Burrough, 1998) If someone wanted to display their data in full, it would require the creation of multiple copies of the same map with different data sets on each, often with accompanying graphs. This approach is very time consuming and exorbitantly expensive, meaning it was close to impossible for independent groups to display graphical data on maps. GIS arose as an attempt to smooth out these difficulties and to allow easy data display at approachable cost.

GIS is centred around the use of…

## 2.2 Existing Mapping Systems

### 2.1.1 Map Storage and Search

Old Maps Online allows the user to search and navigate through a collection of historical maps. (Klokan Technologies GmbH 2013) The user inputs a specific location of interest by either typing into a text box or by pressing on the appropriate area on a blank world map. The website then displays maps relevant to that location. The returned maps do not appear to be in any specific order and original production date information is often absent. This comes across as a big problem for the system and it makes the website difficult to use as you are unsure of what will be returned. The final system will need to return maps in some sensible order to avoid this problem. Although the site has problems, the act of pressing on the map to select a region is a very interesting UI choice and greatly improves the user experience. The idea would likely be straightforward to implement in the project solution, allowing a user to find a map series in a specific location. Rather than navigating through a menu.

Old Maps Online uses a very heavily modified version of Open Street Map with Open Layers.

Old-Maps is a service like Old Maps Online but with more of an emphasis on selling historical maps. Unlike Old Maps Online, Old-Maps displays maps in a chronological series in a navigation menu on one side of the screen that makes maps very easy to navigate through in order. This design choice is really very nice – having a navigation side menu makes the entire process very easy to understand. Navigating around and zooming into an individual map in not an easy task and the website often takes some time to transition between zoom levels. The website doesn’t make it clear when it’s loading and at first it appeared as though the website had crashed – something to avoid. As a whole, this results in a displeasing user experience. The website has taken a series of questionable design choices that will need to be avoided, but an approach like the side navigation menu could well be something to investigate implementing. Unlike Old Maps Online, Old-Maps appears to use bespoke software built on top of ordinance survey data.

The National Library of Scotland also maintains a collection of historical maps. In a similar way to Old-Maps, the NLS has a clickable map that displays a series of relevant maps in a side view box. The NLS approach is different however, using pre-defined highlighted boxes. This approach is not as good as Old-Maps from an aesthetically pleasing point of view and results in a cluttered display, but it may – arguably – make the website a little easier to navigate through. The actual map display is very nice in comparison to the earlier software, selected maps are displayed in a zoomable sub page that allows for easy navigation. Moving between maps is a problem, the only way to navigate back or forward through maps is to move back to the original map and search from the start again. This is a slow process that could easily be improved upon by importing the side box into the map view page. The NLS, like Old Maps Online also uses a version of Open Street Map.

MyHistoricalMap is another website that tries to fill the demand for viewing historic maps, though this one is probably the worst from several standpoints. The website is clunky, difficult to navigate and the map display is not at all clear. The website serves more as an example of what not to do. After some additional investigation, it appears that the service lost development support towards the end of 2018, again highlighting the fact that only aesthetically pleasing and easy to use services are successful. Like Old-Maps, MHM uses data from the ordnance survey

As is clear to see, all the discussed websites are run using at least a large part of their own bespoke software for displaying maps, even if it is just heavily adapted existing software.

### 2.1.2 Mapping Functions

Moving away from map storage and search,

Liveuamap (Live Universal Awareness Map co 2014) is a web-based live map for tracking conflicts in areas of unrest. The display has a series of coloured polyhedrons differentiating who controls which area of land. The system uses ‘buttons’ overlaid on the map which expand when pressed on to allow the user to very quickly and easily examine a detailed description of the event that took place there without making the display overly complicated. The system is heavily geared to the ‘here and now’ and viewing events from previous days is not a straightforward task, requiring navigation of a series of menus to accomplish. This approach is vastly different from what has been examined previously and it is clear that the creators prioritised usability and aesthetic appeal over anything else. The system relies on a small number of administrators to update the display and users are unable to make their own changes or add their own maps. Further, by making the maps ‘live’ and constantly searching for updates, the website draws an unusually large amount of bandwidth and processor power. The website was nigh on unusable on mobile. This is something to avoid in the final design. The user interface is very pleasing to look at and the interactivity is extremely well done. The use of button overlays is very effective at minimising the amount of information on screen at any one time. Similar to Old Maps Online and the NLS, Liveuamap uses a heavily modified version of Open Street Map. It appears that Open Street Map is something of an industry standard in this field.

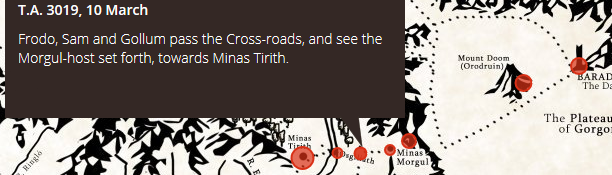
Google’s “My Maps” service (Google 2007 / 2014) allows users to create their own maps by placing layers on top of a blank world map. The user can create these layers by selecting a series of points on the map, saving them to a database and then filling the area within those points with colour. This appears to work in a similar way to the JavaScript SVG Polygon function and seems to be comparable to how Liveuamaps handles its layers. This technique appears to be a standard in many online mapping tools and for good reason, it would be relatively simple to implement such a function. My Maps also allows you to place markers on the map, allowing the developer to hide details until the user requests them and keeping the UI clear – again, like how Liveuamap works. My Maps has no way of connecting a series of maps and so is limited in that capacity.

Looking once more at Liveuamap, the use of map markers on this site is something else to consider in a more focused way. Liveuamap presents its event descriptions in an innovate way, compared to the other examples that have been examined so far.



Each actor taking part in a conflict is assigned a colour and each event is assigned an image from a small collection. This combination of a unique colour and a descriptive image allows a user to very quickly analyse the whole map in significant detail. If the user wishes for a more detailed breakdown, clicking on one of the markers causes additional information to be displayed in a text box and a second box appears on one side of the display offering still further information and links to sources. This minimalist approach is extremely effective at minimising screen clutter and allows only the most important information to be seen at load. Of all the software that has been examined so far, this approach is by far the most effective and will certainly be something to consider in the final design.

Continuing this, the Lord of the Rings Project (Emil Johansson, 2012) offers a similar approach to displaying map markers.



Here, map markers are larger if multiple events take place there. When clicked on, markers expand with an information box giving a breakdown of the details associated with it. This approach is far less effective than liveuamaps approach as there is no fast way to analyse the map, each marker must be clicked on individually to understand the story. This website did offer one useful insight – it uses open layers. The Lotr Project is in many ways similar to the end goal of this project – navigation around arbitrary maps…

# Chapter 3

**System Requirements and Analysis**

## 3.1 System Overview

After the analysis of current trends and standards in the mapping software world, it is time to define the proposed project.

(For a quick glance at the project requirements, view the explicit list on page 15)

‘A Time Traveller’s Map’ will function primarily as an educational aid for children both at home and at school. The software will need to display maps in an aesthetically pleasing and easy to use way and should allow the user to navigate through a series of maps that have been deemed relevant to a given request i.e. a specific location or an event. This navigation should be achieved through a slider and a date and location input box.

The user should be able to upload their own maps and define key features about those maps. They should also be able to connect a series of maps so that they can search look them in order. This map and series should be shareable between users and should be accessible to the creator on multiple devices.

Creators should be able to place markers at important locations by pressing on appropriate locations on the map. They should be able to define information about these markers. The markers should be expandable on click to show this information. The markers should be removable and moveable.

Users should be able to define regions by selecting a series of map points and then give information about these regions. Pressing on a region should display the additional information about it.

Both markers and regions should be minimizable in order to keep the display as clutter free as possible.

Users should be able to upload world events that contain a series of maps with attached markers that make up a ‘story’. These maps should be sharable and should appear automatically in a side scroll box when viewing maps relevant to the events.

Consideration should be paid to user accounts and groups during the design phase in order to maximise the ease of sharing maps and series.

## 3.2 Requirements

### 3.2.1 Requirements Overview

The following section outlines the seven key tenets that will permeate the entire project. These requirements are what should always be referred to in order to ensure the system is staying true to the original aims.

**Ease of Use**

Considering the main target audience of this application is children, ease of use is perhaps the most important concern. Options, instructions, actions and notifications should be easy to understand and intuitive to use. Tooltips and text boxes should be in simple English. Users are not expected to have any expert knowledge to use the service.

**Aesthetically Pleasing**

One of the key points taken from the literature review was that the most successful mapping software was pleasing to look at. Further research into this topic showed that “users can judge a web site's credibility in as little as 3.42 seconds merely on the basis of its aesthetic appeal” (Farah Alsudani 2009) An inescapable fact is that good applications look good and a significant amount of time will need to be spent ensuring that the project is aesthetically pleasing.

**Performance**

As the literature survey has shown, mapping software can be very processor intensive. The system should be geared towards minimising computational complexity and memory use where possible. This is further reinforced by the desire to keep the system as accessible as possible. Important operations should be fast to execute and, in cases where some wait time is expected, a notification should appear alerting the user that the program is still running. Tests dedicated to maximising performance will need to be run during the Testing and Evaluation phase.

**Stability**

The system is expected to be usable at any time. It will need to protect against crash causing errors and invalid inputs. The system should be usable on launch to avoid unnecessary database downtime.

**Platform Independence**

Ideally, the system should be able to run on any machine running any operating system or browser with no noticeable difference between the distribution. This may prove to be unfeasible, but an effort will need to be spent to make the system as cross-platform compatible as is possible. At a minimum, the system should run on Linux, MacOS and Windows operating systems as well as Chrome, Firefox and Edge browsers.

**System Security**

Although the project is highly unlikely to be the target of an attack, it is good practice to be prepared. The system should be resistant to common types of attacks. In 2013, “the average prevalence rate of injectable URL of IPv4 was 5.55%” (Ying - Chiang) so SQL injections should be defended against with the use of sterilised user input boxes and explicit parameterisation. Cross Site Scripting attacks should be resisted where possible. Explicit attention should be paid towards the file upload system. Effort should be spent ensuring that only ‘safe’ file types are uploadable.

**Data Protection**

In order to comply with the European GDPR guidelines (General Data Protection Regulation 2016/2018), the system will need to ensure any data taken from a user is stored safely and in a format that limits vulnerability. Stored user data will need to be minimised, only essential information should be taken to identify the user.

## 3.3 System Architecture Decisions

### 3.3.1 Platform

Throughout this project, the focus has repeatedly returned to the initial decision of wanting to keep the software as easy to access and interact with as possible, and many of the choices made through the course have been with this in mind. To this end, the decision of which platform to base my system on was a relatively straightforward one.

Desktop systems have one major advantage over any other type of platform – no data storage cost the author. This was a significant consideration when deciding on a platform but was eventually discounted for several key reasons.

The first and most important is that each new user would need to download a separate application to their own machine. This might not be a problem if this project was just aimed towards personal use, but that is not the case for this system. Maps are meant to be shared between users and for this to be achieved, the system would need to download copies of every map used onto every PC and the user would need to manually conduct this update. This problem will only grow as the database gets larger. All these problems exist even before considering copyright issues associated with storing protected maps on personal devices.

As a continuation of this problem, it would be very difficult to transfer save states between a user’s devices. Making maps on a computer and then transferring that map series to a laptop would require significant time and effort.

From a more technological perspective, desktop applications are harder to develop platform independent software for - requiring careful use of standards. From a long-term standpoint, if the project were to be expanded to require a GPU to display maps, the system would largely be limited to Java development on the JDK and – as COM2002 taught the year group – developing graphical software in Java can be an excruciating task when aesthetic appeal is a key concern.

A desktop application also suffers the problem of difficulty transferring to a mobile platform. A web application, on the other hand, will be relatively simple to convert to mobile and visa-versa allowing for even greater system reach.

Finally, the problem of bug fixing. If the system ships with a large undetected bug, it would be a huge amount of work to fix. It would also require some sort of software updating ability, something that would require significant research and work to build safely and legally. A web-based bug could be discovered, fixed and updated without the user’s knowledge.

With all the listed reasons, the decision of which platform to use in the initial deployment can be narrowed to mobile vs web-based.

Both options allow for far greater accessibility than a desktop-based alternative and it is difficult to set them apart using only this metric. Because of this, reference back to the core tenets is required to distinguish them. As outlined previously, the most important concern is ease of use and in this respect, a web application is far superior. A mobile application doesn’t offer the finger room to navigate or interact in a highly detailed way and would limit the use of complex functions.

A mobile application would also be breaking the third tenet. Processor requirements to view, load and update maps are high and only good phones will be able to handle this. Testing some of the previously discussed mapping software on an iPhone 6s was not an easy task and forced the user to deal with constant jittering and freezes. It would not be acceptable to demand the newest phones to access the application, especially considering the target audience.

Finally, consider common successful mapping systems like Google Maps, web-based mapping software is very much the industry standard for casual map viewing. That’s not to say mapping software does not exist on desktop or mobile, in fact, there is a great deal of desktop-based mapping software, industrial standard GIS for instance, but these are mainly geared towards professional usage due to the very high cost of such systems. To make a successful, free mapping service in the modern era, it must be web-based.

Because of this, the only acceptable platform would be web-based. In the future, attention could be turned towards creating a limited, processer friendly mobile version.

### 3.3.2 Programming Languages and Tools

Considering the system will be a web-based application, the choice of which programming language to use is limited. The standard front-end development languages are HTML, CSS and JavaScript. These languages are very versatile with powerful libraries and compatible software, and so will be used as a front end.

A typical full stack development project needs some way of communicating with a database and managing a server. Base JavaScript is incapable of this so an alternative server-side scripting language will need to be found. The four big options in this field are PHP, Node.js, Ruby and Python. All four options are generally interchangeable with each other in most domains, but some particularities exist between them. Because of this interchangeability, development speed and ease of use are the key considerations.

### 3.3.3 Mapping Software

As discussed in chapter 2, there are several commonly used from end libraries for displaying geospatial data. The most popular of the surveyed websites are Open Street Map, Open Layers and the Google Maps plugin.

### 

### 3.3.4 Database

Considering this application is going to be web-based, some thought must be put into what type of data storage to use. The main decision to make is whether to use a relational SQL style database or NoSQL.

Data structure is a key consideration when deciding on a database structure. NoSQL style databases are designed for data with little structure that is prone to regular change, whereas SQL databases are designed for rigid data sets. The data layout in this project is very unlikely to change soon, so the structure will be static.

Performance is also something to consider. According to Kumar (Comparative analysis of NoSQL 2015), NoSQL style databases perform queries approximately 40% faster than their SQL counterparts. Despite this, in the same paper Kumar outlines that this effect is only noticeable when considering thousands of queries at once and the speed difference will not be felt on my application. This may change at some point in the future if the software expands and demands a greater number of queries.

Ease of development is another concern. SQL style databases have a single de facto standard language that is well documented. NoSQL style databases do not have this privilege, instead relying on varying competing standards with limited documentation and tools. Yishan Li (A performance comparison of Databases 2014) and Lewitt (2010) agree saying “there is a lack of support tools to help.” Due to the limited time frame available to develop this system, attention needs to be paid to development speed and a well-documented style will result in far faster development.

Experience is a lesser concern, but it is still something to consider. The author has a deal of experience developing applications using SQL style databases and that experience will transfer into rapid development. It would be fair to expect NoSQL based development to be considerably slower, requiring significant time to learn a new system.

Considering that the key advantages of NoSQL style databases, big data scalability, frequent structure changes and performance will have very limited effects on the system, the only real differentiating factors are ease of development and experience. It is likely that development will be far faster using an SQL style database. Because of this, the system will be built with an SQL backend.

### 

### 3.3.5 DBMS and Mapping Data

One thing to consider when choosing a database flavour is whetherthe chosen service can store and use data in an efficient way. The project, as mapping software, will rely heavily on longitude and latitude values and because of this, a database that can store and with libraries to manipulate data in the form [(X, Y)] will be heavily preferred.

This requirement immediately discounts the use of MySQL, as the service offers no support for geospatial data. MySQL is also not appropriate because it only offers library support for calculations based in Euclidian geometry. (Openlife.cc, 2016) This problem could lead to accuracy inconsistencies close to a planet’s poles and between data sources. OpenSQL and SQLite both suffer from a similar problem, being unable to store and use longitude and latitude with accuracy.

Because of this, one of the few remaining popular SQL style databases to consider is PostgreSQL. This style is different in that it offers two very popular libraries for Geospatial storage and manipulation – Earthdistance and PostGIS which, combined offered a huge array of functions. Both libraries are well documented and have significant user bases, allowing for fast bug fixing and development rates.

Because of this, the system will use PostgreSQL.

## 3.4 Anticipated Development Problems

Considering the specificity of this project, tools and libraries have a very limited number of users and code bases that are relatively untested when compared to standard and commonly used libraries. Having conducted some research, it is fairly standard for some mapping libraries to have less than ideal documentation and the implementation may require some significant level of experimentation to get these tools to work correctly. This problem is likely to be compounded by the authors inexperience working in this field. Productivity will need to be measured and a strict time table will need to be adhered to. In the event of a continuous stubborn roadblock, it may become necessary to remove or delay features in order to stick to completion schedules.

Continuing this point, testing may also be a problem. Specialised libraries often aren’t completely compatible with standard testing tools. Care will need to be taken during testing to ensure particularities in the code base are not causing unexpected - or worse - unreportederrors.

For a less software-based consideration, copyright is going to be something to pay attention too. Many maps are owned by the map makers and are not licensed for public dissemination. The website will need to do what it can in order to avoid legal trouble, or at a minimum, pass the legal burden to the user.

In general, effort to minimise the number of bugs in the code makes for standard good coding practice but, as always, some bugs will slip through the cracks. To combat this, regular test processes should be run, followed by a large final test to ensure the system ships without any known bugs. Outlines for this testing can be found in section [**Testing Section**]

## 3.5 Ethical Considerations and Pre-Implementation Changes

Originally, this project was intended as an educational aid for teachers and children. This is still a key focus, but after some advice and consideration, some aspects have had to change. The amount of data that was to be stored on the system raised some important ethical concerns. Storing large collections of data from children brings additional unnecessary risks and is something to be extremely wary of in general. Because of this, the required information has been slimmed to the bare minimum - no delicate personal information will be stored at all. The only details to be requested on sign up will be an email address, username and suitable password. Usernames will be the only public information available to users.

Further, cyberbullying and abuse are things to consider. It is conceivable that users could upload offensive images in place of maps. To combat this, users should only be able to share maps with people whose usernames they know and have whitelisted. This should prevent unsolicited images being shared to a wide audience.

Additionally, data management is something else to consider. Minimising stored data in the way described above has the added benefit of lowering system scaling costs as there is less data to secure and handle. Further, the process of database setup and management is streamlined significantly and should lead to quicker initial preparation, bug fixing and general database maintenance.

## 3.6 Specific Requirements

Below is an explicit list of requirements for the system - a checklist for development. Requirements are ranked by priority – ‘Maximum’ is necessary for a viable minimum product.

**M. Map Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| Identifier | Requirement | Priority | Exp. Difficulty |
| M1 | User can upload permanent maps | Maximum | Low |
| M2 | User can view selected maps | Maximum | Low |
| M3 | User can delete maps | Maximum | Low |
| M4 | User can edit map details | Maximum | Low |
| M6 | User can define a location for their maps | High | Low |
| M7 | User can define other details for their maps | High | Low |
| M8 | User can view other people’s maps | Medium | Medium |
| M9 | Relevant maps should be displayed on start up | Low | Very High |
| M10 | User can zoom the map in and out. | Low | Very High |
| M11 | User can search for locations by clicking on the map | Low | High |
| M12 | User can add maps directly from the internet | Low | Unknown |

**E. Map Events**

|  |  |  |  |
| --- | --- | --- | --- |
| Identifier | Requirement | Priority | Exp. Difficulty |
| E1 | User can add events to a map with the mouse | Maximum | High |
| E2 | User can view events on a map | Maximum | High |
| E3 | User can remove events from a map | Maximum | Medium |
| E4 | User can define a name for an event | Maximum | Low |
| E5 | User can define an event description | High | Low |
| E6 | User can define and start and end point to events | High | Medium |
| E7 | User can view an expanded details box for an event | Medium | Medium |
| E8 | User can view the details box with a mouse click | Medium | Medium |
| E9 | User can toggle event appearance on the map | Low | Low |
| E10 | User can add custom map events | Low | Medium |

**W. World Events**

|  |  |  |  |
| --- | --- | --- | --- |
| Identifier | Requirement | Priority | Exp. Difficulty |
| W1 | User can view a collection of world events | High | Medium |
| W2 | User can create world events | High | Low |
| W3 | User can delete world events | High | Low |
| W4 | User can add start and end points to world events | High | Low |
| W5 | User can edit world event details | High | Low |
| W6 | User can select a world event and get relevant maps | Low | Medium |
| W7 | User can include links to additional information | Low | Medium |

**T. Time Line**

|  |  |  |  |
| --- | --- | --- | --- |
| Identifier | Requirement | Priority | Exp. Difficulty |
| T1 | User can view the time line | Maximum | Low |
| T2 | User can select required year with the timeline | Maximum | Medium |
| T3 | Current year is visible somewhere on screen | Maximum | Low |
| T4 | User can navigate by defined time periods blocks | High | Medium |
| T5 | User can navigate time periods with a text input box | High | Medium |
| T6 | Upcoming and recent world events are visible | High | Medium |
| T7 | Upcoming and recent map transitions are visible | High | High |
| T8 | Upcoming and recent map events are visible | Low | Very high |

**U. User**

|  |  |  |  |
| --- | --- | --- | --- |
| Identifier | Requirement | Priority | Exp. Difficulty |
| U1 | User can create an account | Optional | High |
| U2 | User can sign in and out of their account | Optional | High |
| U3 | User can authorise their account through email | Optional | Medium |
| U4 | User can create a map linked to their account | Optional | Low |
| U5 | User can create a map series | Optional | Low |
| U6 | User can link a map to their account | Optional | Low |
| U7 | User can delete maps from their account | Optional | Low |
| U8 | User can share their maps with other users | Optional | Unknown |
| U9 | User can whitelist other users | Optional | Unknown |

# Chapter 4

**System Design**

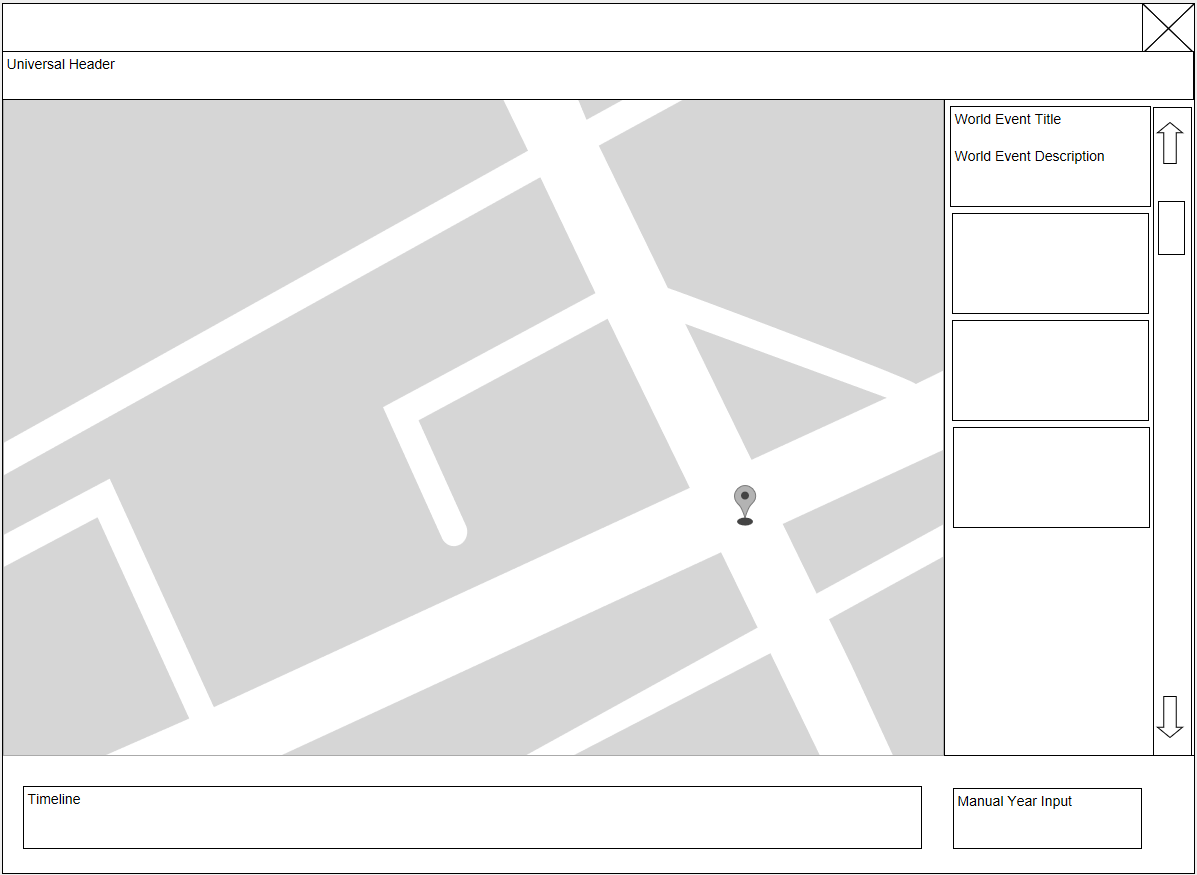
In this chapter, I will be designing some initial mock-ups for the website’s design. I will be making designs for all major features and will use them as a guide going forward. I fully expect many of these designs to change slightly throughout the course of development and in the event that any requirements change.

After the GUI design, I will create the designs for the database.

## 4.1 User Interface Design

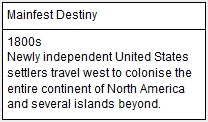
### 4.1.1 General Layout

When designing the basic layout for my project, I aimed for a minimalist approach with most of the functionality hidden but still easy to figure out. To that end, the main screen will just display the most basic features of the system with further functions hidden in minimised menus in the header. The main map will take up the majority of the screen with only a small area for the time line. World events will appear in a scrollable side box that the user will be able to hide should they wish for more map display.

****

### 4.1.2 World Events

World Events are user created events that will have a series of maps connected to them. By clicking on them, a different user will have the option of exploring the topic in detail or finding relevant maps in the database. This is an example of a World Event.



World events will have four key features attached to them, a start date, an end date, a location where the event took place and a type which will be used to decide on which field in the appropriate database table to look through, i.e. ‘Global’, ‘National’, ‘Local’ etc. With this information, I will be able to retrieve relevant maps from the database that the event creator may not have seen before. I would like to be able to share these events publicly through the website so that users will be able to see a large collection of maps for a large collection of events.

World events that are close to the current time-period that the user is looking at should appear in the right hand scroll box. For example, if the user is looking at a series of maps from the British conquest of India, they will also be prompted to view a map series of the Seven Years War on mainland Europe, if such an event has been created before.

### 4.1.3 Timeline

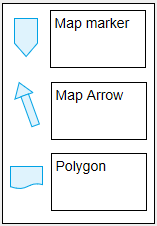
- To Do -

### 4.1.5 Map Annotation

Each layer will have a number of map events attached to it. A map event is a clickable occurrence on the map that will expand to give additional information.

I have decided on three UI map events that will be a part of the initial minimum viable product:

* A *Polygon* – Polygons are shaded, multi point events that show an area.
  + The territory held by a group
  + An area that has been flooded
* An *Arrow* – An Arrow is an occurrence that takes place between two points on the map and represents movement.
  + The movement of an army.
  + The spread direction of floodwaters.
* A *Marker* – A Marker is a single point on the map, a location of interest.
  + The site of a battle.
  + The location of an important meeting.

If I have time at the end of this project, I would like to investigate the possibility of adding custom user defined events.

To do this, I will use a drop-down menu like this.

**Bibliography**

http://whc.unesco.org/en/list/85