

THE UNIVERSITY OF TEESSIDE
SCHOOL OF COMPUTING AND MATHEMATICS
MIDDLESBROUGH
CLEVELAND TS1 3BA

TEES VALLEY FROM THE AIR

***EDITED ONLINE COPY*(no appendix)**

Full copy available on request

Final Report

March 2004

John Laing

BSc (Hons) Computer Studies

ABSTRACT

This project presents a website that allows the user to view aerial photography of the Tees Valley area. Information concerning the places of interest in the area is also available as well as a look at the history of the towns in the area.

There are two parts to the website, the web interface and the underlying database. The database, created using MYSQL, stores the information relating to the places of interest, whilst also handling the aerial images, aerial views and data for the aerial images search.

The web interface, created using Dreamweaver MX, allows the user to search for and view the aerial images. Other features allow the user to navigate the images and view road and overview maps. The user can also search the database for information on attractions. PHP is used to connect the interface to the website to allow for the data to be displayed. Historical information can also be accessed as well as the chance to sign up for a future membership.

To maintain the database and to keep the data secure, an administration section will be present within the website. This area is password protected using JavaScript. The attractions information held within the database can be accessed and edited accordingly by the administrator.

The website is a prototype that meets the requirements that were stated during the analysis of the project. The town of Marske is fully covered with aerial images and surrounding functions and the database contains enough data to allow all functions to be demonstrated successfully. There is scope for further features to be added if a full version of the website was to be produced.

CONTENTS

| | |
|--|-----------|
| THE UNIVERSITY OF TEESSIDE..... | 1 |
| CLEVELAND TS1 3BA..... | 1 |
| TEES VALLEY FROM THE AIR..... | 1 |
| *EDITED ONLINE COPY*(NO APPENDIX)..... | 1 |
| FULL COPY AVAILABLE ON REQUEST..... | 1 |
| ABSTRACT..... | 2 |
| CHAPTER 1: INTRODUCTION..... | 5 |
| CHAPTER 2: METHODOLOGY..... | 7 |
| 2.1 GENERAL DEVELOPMENT METHODOLOGY..... | 7 |
| 2.2 RESEARCH METHODOLOGY..... | 7 |
| 2.3 ANALYSIS METHODOLOGY..... | 8 |
| 2.3.1 UML USE-CASE DIAGRAMS..... | 8 |
| 2.4 DESIGN METHODOLOGY..... | 8 |
| 2.4.1 WEBSITE DESIGN..... | 8 |
| 2.4.2 DATABASE DESIGN..... | 8 |
| 2.5 IMPLEMENTATION METHODOLOGY..... | 9 |
| 2.6 TESTING METHODOLOGY..... | 9 |
| CHAPTER 3: RESEARCH..... | 10 |
| 3.1 TEESS VALLEY..... | 10 |
| 3.2 GETMAPPING..... | 11 |
| 3.2.1 AERIAL IMAGES..... | 11 |
| 3.2.2 SURROUNDING FUNCTIONS AND FEATURES..... | 12 |
| 3.2.3 SEARCH FUNCTION..... | 12 |
| 3.3 MULTIMAP..... | 13 |
| 3.3.1 AERIAL IMAGES..... | 13 |
| 3.3.2 SURROUNDING FUNCTIONS AND FEATURES..... | 14 |
| 3.3.3 SEARCH FUNCTION..... | 14 |
| 3.4 RESEARCH CONCLUSION..... | 15 |
| CHAPTER 4: ANALYSIS..... | 16 |
| 4.1 WHO WOULD USE THE WEBSITE?..... | 16 |
| 4.2 WHAT WOULD THE USER BE LOOKING FOR?..... | 16 |
| 4.3 HOW WOULD THE USER SEARCH FOR INFORMATION?..... | 17 |
| 4.4 TECHNOLOGIES TO BE USED IN IMPLEMENTATION..... | 17 |
| 4.4.1 IMPLEMENTING THE WEBSITE INTERFACE..... | 17 |
| 4.4.2 IMPLEMENTING THE DATABASE..... | 17 |
| 4.4.3 CONNECTING THE DATABASE TO THE WEBSITE..... | 18 |

| | |
|---|-----------|
| 4.5 FUNCTIONALITY..... | 18 |
| 4.5.1 USE-CASE SCENARIO 1..... | 18 |
| 4.5.2 USE-CASE SCENARIO 2..... | 19 |
| 4.5.3 USE-CASE SCENARIO 3..... | 19 |
| 4.5.4 USE-CASE SCENARIO 4..... | 19 |
| 4.6 OTHER INFLUENCING FACTORS | 20 |
| 4.6.1 SECURITY..... | 20 |
| 4.6.2 MAINTAINABILITY..... | 20 |
| 4.7 PROTOTYPE REQUIREMENTS..... | 20 |
| CHAPTER 5: DESIGN..... | 21 |
| 5.1 DESIGN OF THE WEBSITE..... | 21 |
| 5.1.1 DESIGN OF AERIAL IMAGES SECTION..... | 23 |
| 5.1.2 DESIGN OF ATTRACTIONS SECTION..... | 27 |
| 5.1.3 DESIGN OF HISTORY SECTION..... | 27 |
| 5.1.4 DESIGN OF ADMINISTRATION SECTION..... | 27 |
| 5.1.5 DESIGN OF THE HELP PAGE..... | 28 |
| 5.2 DESIGN OF THE DATABASE..... | 28 |
| CHAPTER 6: IMPLEMENTATION AND TESTING..... | 30 |
| 6.1 IMPLEMENTATION OF THE EVOLUTIONARY VERSIONS | 30 |
| 6.2 VERSION ONE..... | 30 |
| 6.2.1 CREATING TEMPLATES | 30 |
| 6.3 VERSION TWO..... | 31 |
| 6.3.1 FUNCTIONALITY OF MAIN WEB PAGES | 31 |
| 6.3.2 IMPLEMENTATION OF AERIAL IMAGES..... | 32 |
| 6.4 VERSION THREE..... | 34 |
| 6.4.1 IMPLEMENTING FORMS | 34 |
| 6.4.2 IMPLEMENTING THE MEMBERSHIP SECTION..... | 35 |
| 6.5 VERSION FOUR..... | 37 |
| 6.5.1 IMPLEMENTING THE DATABASE | 37 |
| 6.6 VERSION FIVE..... | 39 |
| 6.6.1 ALLOW THE USER TO VIEW ATTRACTIONS | 39 |
| 6.6.2 ALLOW USER TO USE TEXT BOX SEARCH FOR AERIAL IMAGES | 40 |
| 6.6.3 DISPLAY AERIAL VIEWS..... | 40 |
| 6.7 VERSION SIX..... | 40 |
| 6.7.1 DISPLAY AERIAL IMAGES AND POP UP WINDOWS | 40 |
| 6.7.2 IMPLEMENT NAVIGATION COMPASS AND LINKS..... | 41 |
| 6.8 VERSION SEVEN..... | 42 |
| 6.8.1 ALLOW ADMINISTRATOR TO EDIT ATTRACTIONS TABLE | 42 |
| 6.8.2 ADDING ATTRACTION..... | 42 |
| 6.8.3 DELETING ATTRACTION..... | 43 |
| 6.8.4 UPDATING ATTRACTION..... | 44 |

| | |
|---|-----------|
| 6.9 VERSION EIGHT..... | 45 |
| 6.9.1 ALLOW ADMINISTRATOR TO LOG IN | 45 |
| 6.10 TESTING OF THE PROTOTYPE WEBSITE..... | 46 |
| CHAPTER 7: CONCLUSION..... | 47 |
| CHAPTER 8: RECOMMENDATIONS..... | 49 |
| CHAPTER 9: REFERENCES..... | 50 |
| 9.1 INTERNET REFERENCES..... | 50 |
| 9.2 BOOK REFERENCES..... | 50 |

CHAPTER 1: INTRODUCTION

At the beginning of the Millennium year, a company called Getmapping [1] produced the millennium map. This map was unlike anything created before it, as it was pieced together using birds eye view aerial images. It became hugely popular as it allowed people to view where they lived from a different perspective. Since the creation of the website, from where you can view this map, the actual product has been expanded and is accessible by other forms of media such as books and games.

My project idea came about whilst I was viewing the Getmapping website. Due to the grand scale of the website, it was apparent to me at an early stage that I would be unable to recreate a website of that size. I realised my best approach would be to concentrate on aerial images of a particular place. I then decided to focus on the Tees Valley area.

I decided to introduce a tourism element to the site. Users will be able to view information on places of interest in the area. In addition, a section that covers the history of the area is also included.

The main objectives of this project are as follows:

To produce a website that will allow users to search and view aerial images and attractions information. Historical information can also be accessed.

To ensure the database data is secure, and accessible only by an administrator.

To ensure the website is user friendly and easily maintained.

Developing a website that will handle database information generates a number of issues that have to be resolved:

The security of the information must be taken into account. It would be inappropriate for a user to be able to access the database information as it may lead to required data being lost or altered. Also the right format of data must be held within the database.

This report will describe in detail the methodology, research, analysis, design, implementation and testing of the website and databases. It is structured as follows:

The methodology section introduces the methods I used throughout the various stages of the development process. Also it outlines the reasons for why they were chosen.

The research section is a detailed discussion on both Getmapping and Multimap websites. The discussion will cover aspects such as how the aerial images are displayed, and surrounding functions.

The analysis section will cover the prototypes requirements and how they were obtained and specified. A discussion on the technologies to be used to implement the prototype website is also documented.

The design section covers both the design of the website interface and the database.

The implementation and testing describes how the website was produced and how the testing was carried out.

The conclusion evaluates the success of the prototype website with regards to the initial specification and requirements.

The final chapter discusses any future recommendations and possible extensions to the website.

The appendices contain extra material that supports the main body of the report.

CHAPTER 2: METHODOLOGY

This chapter will discuss the various methods used throughout the stages of the development of the prototype website.

2.1 General Development Methodology

For the development of my prototype, it was decided that no methodology would be employed fully until the implementation stage. The John December methodology [2] was used as a guideline for the various stages of the project building up to the implementation. It was decided not to follow the processes and elements of the method as in some cases, they were irrelevant to the development of my website as they focused on the production of a fully implemented website.

When it comes to the implementation stage of the prototype, the Evolutionary Delivery development methodology [3] will be used. This allows for the implementation development to be carried out in a number of incremental versions.

Using this method, a very basic website that offers very little functionality would be created. This would then be tested before more functions were implemented. Once these additions were approved, a newer version of the website would be developed and so on until the final prototype was finalised.

By using this method, it allowed for any errors to be found at the earliest possible stage. If the implementation of the prototype was carried out in one go, any resulting errors would be difficult to trace and it would result in a poorly constructed website.

2.2 Research Methodology

The purpose of the research was to get a general idea of how aerial images are used on the web. Looking at the two websites that formed the basis of my research, I looked for the following:

How the aerial images are displayed
Functions available that work along side the aerial images
Search Options

2.3 Analysis Methodology

An early analysis of the project during the planning stage was carried out to determine whether the project was viable or not. This involved contacting Getmapping, who host the aerial images, to ask if I could use the images for the project. Once permission was granted, I could then determine a path forward for the project, including making decisions on what technologies to use for the best possible outcome. A questionnaire was produced at the analysis stage so that information could be collected for a particular section of the website. A discussion on the target user and the website requirements are also documented.

2.3.1 UML Use-Case Diagrams

The Unified Modelling Language (UML), in the form of use-case diagrams [4] was used to simplify the task of understanding how the website will work in terms of user control and navigation.

2.4 Design Methodology

The design covers both the website interface and the database.

2.4.1 Website Design

The design of the website and user interface was achieved by creating a collection of storyboards and navigation charts. Each design document outlines the basis of the design for each section of the website. The navigation chart organises the structure of the files of the website.

2.4.2 Database Design

The database will be a collection of stored data that will be used for different functions on the website. A decision on which data should be stored was based upon the findings in the research and analysis stages. Once the information had been decided upon, the tables that will handle the information were designed.

2.5 Implementation Methodology

As mentioned earlier in the Methodology chapter, the Evolutionary Delivery development method will be used.

2.6 Testing Methodology

Testing is carried out after each version of the website is implemented. The testing is to ensure that all functions and links within the website work.

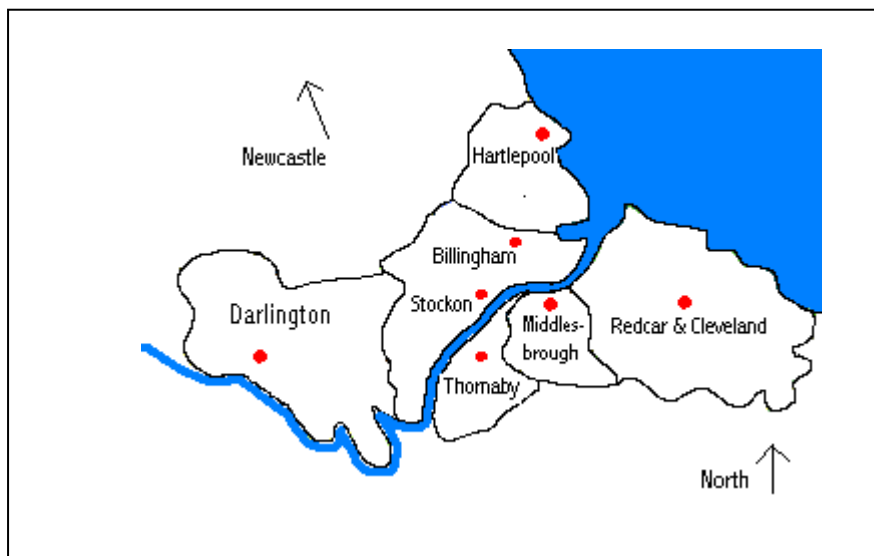
CHAPTER 3: RESEARCH

The basis of the research was on two websites Getmapping and Multimap [5], both of which display aerial images.

The research is a discussion of how these websites display the images to the user, and any useful functions. Although in depth research on both websites was carried out, any information found to be irrelevant to my prototype website will not be discussed.

3.1 Tees Valley

As I mentioned in chapter 1, my prototype website will focus on the Tees Valley area for a number of reasons. It's my local area so I already have background knowledge of the towns and attractions. The area is made up of large towns and small villages, which gives me a number of options when it comes to deciding what aerial images I will use for my prototype.



The above diagram displays the districts and towns of the Tees Valley area.

3.2 Getmapping

Get mapping are the creators of the Millennium Map, the most comprehensive and detailed aerial photographic survey of the UK. Getmapping offers home and business customers the opportunity to purchase products such as aerial maps, digital photography and aerial photo prints

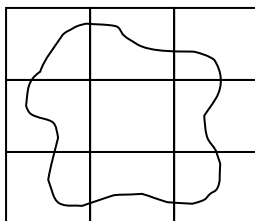
Since first appearing on the web, the website has changed in various ways over the past three years, most noticeably on the marketing side as they try to expand to a larger audience.

Aerial images are supplied to the user in two ways. The first way being allowing the user to view sample aerial images that can be purchased. The second method in which the images are displayed is within a membership section, named imagexpress.

3.2.1 Aerial Images

Firstly, I will discuss the sample images that are available to users who are not members of the imagexpress section of the website.

The images are small in size and are viewable at different scales. The image quality is not the greatest although housing and buildings can be clearly recognised. Each individual image is made up of nine smaller images. An example is shown in fig 3.1



3.1

Each image is overlapped with the surrounding images to allow for continuity whilst roaming. For example, a school is visible to the right hand side of an aerial image. The user then moves east once, and the school is still visible but this time within the centre of the image. An example of overlap is shown in fig 3.2

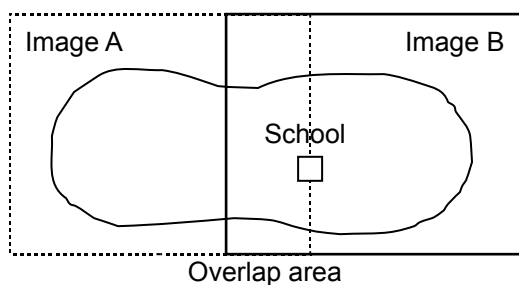


Fig 3.2

The aerial images found within the imagexpress section work on the same principal of the sample images. There are a number of telling differences though; the images are of a higher quality with more detail being shown. They are also larger in size allowing for a greater area of ground to be displayed. I feel the price of the membership to access these images is too costly although there is a pay-per-view option. Personally, I feel the site would benefit much more by allowing the user free access to all functions.

3.2.2 Surrounding Functions And Features

A simple roam function is used in conjunction with the images on the website. This allows the user to navigate via the four main points of a compass. Within imagexpress, the user can also move diagonally which allows for faster navigation.

Another function that increases the interaction between the user and the aerial image is the zoom feature. This allows the user to view an image from varied distances. Whilst viewing the sample images, the zoom function has an extra purpose. Each zoom setting is based on the different sized images that are available for purchase. The image that is purchased is of the same quality as those found in the member section.

A great feature that is found only in the imagexpress section is the road map display. The problem with viewing aerial images of an area you have no previous knowledge, is that you are unsure of where and what your looking at. The road map that is displayed alongside the aerial image ensures that the user is fully aware of the area they are viewing.

Another feature within the imagexpress area allows the user to view a larger scaled aerial image or road map image individually.

3.2.3 Search Function

Getmapping employs two search options, a quick search and an advanced search.

The quick search option requires the user to input the postcode of the area they wish to view. When a valid postcode is entered, a list of links to all address located within that postal area are displayed.

The advanced search allows for more detailed information to be entered:

Street Name
Place name
Motorway
Ordnance Survey coordinate

The options allow different types of data to be entered. For example, a member of the military would use the Ordnance Survey coordinate search option, as that is their commonly used method of locating an area.

As many of the search options above require information that is likely to appear in more than one area of the country, the website displays back to the user all places that match the data input by the user. I like this function as for example; it allows me to type in Stockton instead of the full name Stockton-on-tees. It doesn't force the user to type in the exact details which I always feel is a benefit to any website.

3.3 Multimap

Multimap is in partnership with getmapping so the aerial photography displayed within this website is from the millennium map. There are many functions on the website that are irrelevant to my project website such as traffic information and hotel bookings. Hence, I will not discuss these features.

All functions within the website are fully available to the user without the need of a membership.

3.3.1 Aerial Images

The first thing that was noticeable whilst beginning the research of this website was that the aerial photography wasn't the focus of the website unlike Getmapping. I found that the road maps were the central focus of the site with the aerial images being an extra feature.

The aerial photography is of the same quality as the sample images found on getmapping. A recognisable difference between the aerial photography found on both websites is that Multimap images are a single image and are not made up of smaller individual pictures.

The aerial images are available in small and large sizes, each displaying the same level of detail.

3.3.2 Surrounding Functions And Features

Although the road map feature is the main focus of the website, it is linked with the aerial photography. The user can switch between the image and roadmap and vice versa. A function called 'overlay' that allows the user to view both the aerial image and roadmap together within one image is an excellent feature. When selecting this feature, the road map related to the current aerial image is transparently placed on top of the aerial picture. When the user moves the mouse cursor over the image, the road map is highlighted within the aerial image, allowing both to be viewed together.

The zoom function is in place for both the road maps and the aerial images. There are three ways to zoom. The first method is to click the 'plus' and 'minus' buttons (fig 3.3). The second method is to click on one of the zoom bars located between the plus and minus buttons. The third option is to select an image scale from the drop down menu that is available (fig 3.4)



fig 3.3

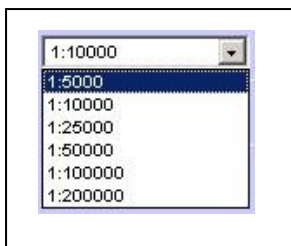


fig 3.4

3.3.3 Search Function

As with Getmapping, the quick search is the postcode option. An advanced search employs two other methods:

Street search

Town search

Until recently, you could use the street search to specify a house or building number but due to the price of the underlying data that powers the service, only the street name can be entered.

Unlike many search options that match the data input by the user and then display all results that contain that information, Multimap does not function this way. The full town name has to be entered for the search to work.

An example, my hometown is '*Marske-by-the-sea*' but if I typed in '*Marske*' within the search box I would be directed to a town named '*Marske*' which is located in another area of the country. I feel this is the only downfall of the website as it can cause confusion to the user.

3.4 Research Conclusion

Having researched both websites, I've found Multimap to be of most use. Getmapping is based on more on the marketing side of the images and access to most areas requires a fee, which I imagine is off putting to many users. Multimap on the other hand allows the user to access all information based on the aerial photography which encouraged me as a user to stay with that website.

There are a number of features on both sites that I have picked up on and I feel they could be implemented into my website but in an appropriate manner. For example, letting the user select which aerial image they wish to view I feel is a very good feature. When the website controls what you view, instead of allowing the user to decided, It can be off putting.

CHAPTER 4: ANALYSIS

This chapter discusses the analysis of the target user for my prototype website. The website requirements and tools used for implementation are also covered in this chapter.

4.1 Who would use the website?

From previous research of existing websites which host aerial images, its difficult to define one particular target user other than the administrator.

As the prototype website focuses entirely on the Tees Valley, its apparent that residents of that area would be more suited to using the website although the location of the user is not an important issue. This is due to the inclusion of tourist information, which may attract users who are planning to visit the area. An analysis table [5] was produced to aid in the analysis of users whom may be interested in using the prototype website. This table can be found in appendix B.

4.2 What would the user be looking for?

Viewing the website will enable the users to view an area of Tees Valley from above. To coincide with each aerial image, there will be a road map that will act as a guide for the user. For users who are unfamiliar with the area, there will be an additional overview map that pinpoints where the user is looking in relation to the rest of the town.

The second objective of the website is to advertise to the user the places of interest [6] in the area. The user can select which information they wish to view from a number of options, thus removing irrelevant data.

The user will be able to read a brief history [7] of the towns in the Tees Valley area.

4.3 How would the user search for information?

To aid the analysis of the website, a questionnaire based upon search options found within the Internet was produced. A sample questionnaire and results chart are documented within Appendix C. Copies of the questionnaire were handed out to two groups of people:

Students with internet access

People without computer access

The reasoning behind collecting data from these two groups of people was that the students would state which functions they know would be best suited to the website. The non-computer users would state which function they would like to see. The results of each group when processed together should result in the most efficient search functions being implemented. The search functions in question are those that will be used to allow the user to search for available aerial photography.

4.4 Technologies to be used in implementation

For the implementation of the prototype website, a decision had to be reached on how to develop both ends of the website, the interface and the database

4.4.1 Implementing the Website Interface

Creating the HTML coding for the web interface can be carried out via two methods. The first method is to generate the coding manually within notepad. This method is very time consuming and is best used for only small editing of code.

The chosen method and the second option available is Dreamweaver. This will allow for me to concentrate on the design of the website as the HTML coding is automatically generated relating to the design. Any extra code can be input within the coding window. The scripting language JavaScript will be used to create the necessary pop up windows.

4.4.2 Implementing the database

There were three technologies available to me for the implementation of the databases. The three choices were Microsoft Access, SQL server and MySQL. My prior knowledge of MYSQL aided in the decision to use that technology. Access was not an option as it allows only a small number of simultaneous connections and its too slow. SQL was an available option but MYSQL is more closely integrated with PHP, which will be used to connect to the database

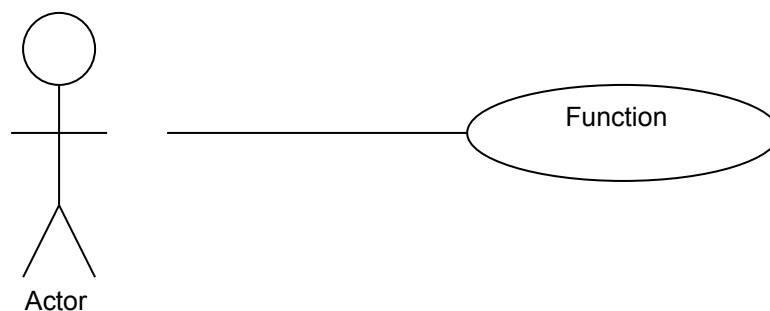
4.4.3 Connecting The Database To The Website

To enable the data stored inside the database to be displayed within the website, I will need to incorporate a scripting language. The two options available were Hypertext Pre-Processor (PHP) and Microsoft's Active Server Pages (ASP). As mentioned in the previous section, PHP has a close integration with MYSQL. Also there are many functions within PHP that are not possible using ASP such as dynamic web pages and email.

4.5 Functionality

As outlined in Chapter 2, section 2.3, it's important to determine what functionality needs to be made available to the user. A series of use-case diagrams were used for this purpose. This places the user as the central actor with links to the functions he/she could carry out within that scenario.

An example of a use-case diagram is found below:



The use-case diagrams produced for this report can be found in appendix D. The following is an outline of these use-case scenarios.

4.5.1 Use-Case Scenario 1

Actor: Administrator

Use-Cases: Logging In

To enable the database to be edited in any way, the administrator must log in so that he/she can interact with the database.

When the administrator comes to log in, a pop up text box will be displayed. If the administrator enters the correct password, he/she is directed to the admin main menu page. If the wrong password is entered, the admin is directed to an error page that contains a link back to the password page.

4.5.2 Use-Case Scenario 2

Actor: Administrator

Use-Cases: Logged In

Once logged in, the administrator has full access to the attractions table within the database. From here, the administrator can add, delete and update the required information. It is a fast and secure way to maintain the information.

4.5.3 Use-Case Scenario 3

Actor: User

Use-Cases: Select town, Select View, View aerial image

This scenario is based on the user wanting to view a particular aerial image. The user must first navigate to the aerial images section. Upon visiting this page, a map and text box is displayed giving the user two methods of which to search by. The map method allows the user to select one of the districts displayed on the map, and then in turn select a town within that district. Selecting a town takes the user to the aerial views page.

If the user wishes to use the text box, they must enter the name of a town. If that particular town is found within the database, a link to the aerial views page is displayed.

The views page consists of a collection of aerial images of the town the user selected. The purpose of these views is that it allows the user to select which area they wish to view first. Each view gives a short description of any roads or buildings located in the aerial image and a link to the page that contains the image. By selecting the view of their choice, the user is then taken to that page.

4.5.4 Use-Case Scenario 4

Actor: User

Use Cases: Select attraction, View attraction information

For the user to view the attractions in Tees Valley, they must first visit the places of interest section of the website. Within this section the user can search for the desired information either by attraction type, location or name. The relevant information is then displayed to the user.

4.6 Other Influencing Factors

There were two key factors in making sure the data held within the database is secure and maintainable.

4.6.1 Security

As the website has no register function, meaning that anyone can browse the website, the information stored in the database is vulnerable. It was decided that the use of an administration section would solve this problem.

This section of the website will be password protected to ensure only the admin can edit the information stored within the database. This will keep the data safe and manageable.

4.6.2 Maintainability

The data held within the database must be easy to maintain. In order to achieve this, the administrator must be able to access and edit the required information, without any changes being made to the coding or structure of the website.

4.7 Prototype Requirements

The requirements of the project prototype website are as follows:

To produce a website which will allow the user to search for and view a towns aerial images. The user must also be able to navigate the town's aerial images and have access to a road map and overview map.

To allow the user to search and view places of interest and to learn a little local history.

Allow the user to sign up for a membership to the website.

A help page which will guide can guide the user through each of the main functions within the website.

To create a database which will store the required data for the functions within the website.

To ensure the databases data is secure and updateable only by an administrator, by creating an administrative section from where changes can be made.

To ensure the website is user friendly and easy to navigate.

CHAPTER 5: DESIGN

Having specified the system requirements, the design stage is the following process that leads towards the implementation.

5.1 Design Of The Website

To produce an easy to use prototype website, a list of design requirements were produced. They are:

- To ensure all sections of the website are interlinked.

- Each web page within each section must be accessible from another page within that section with the use of no more than two links.

- Page titles should be used to indicate to the user what information they are currently viewing.

To incorporate each function that was stated within the requirements section of the analysis chapter, a number of pages must be created. These pages are:

- A home page.

- A collection of search pages.

- A number of pages to display the required information the user searched for.

- Pop up pages within the aerial images section.

- A login page to allow the administrator to access the database information.

- Admin pages from where the database can be updated accordingly.

- Members sign up page.

- A help page.

The first stage of design was to decide on a layout for the user interface. The layout had to incorporate the following:

- A title image

- A static menu with quick links

- A dynamic area in which data will appear

- A page title

- A footer

See fig 5.1 for a storyboard of the layout

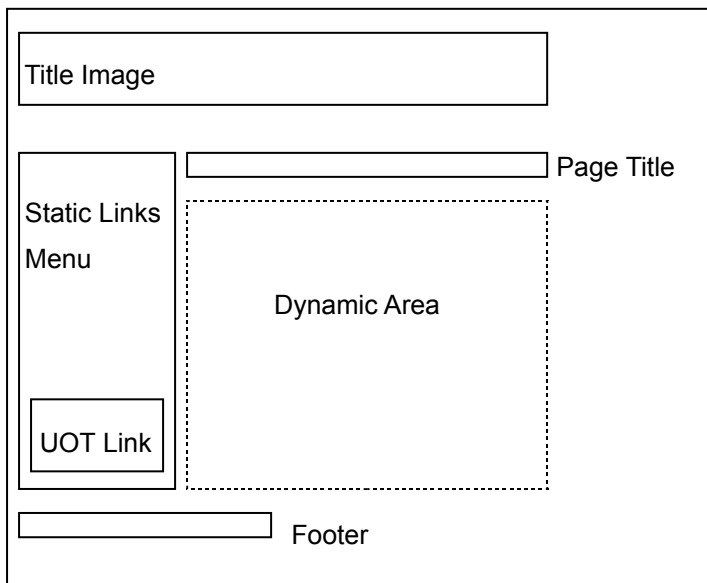
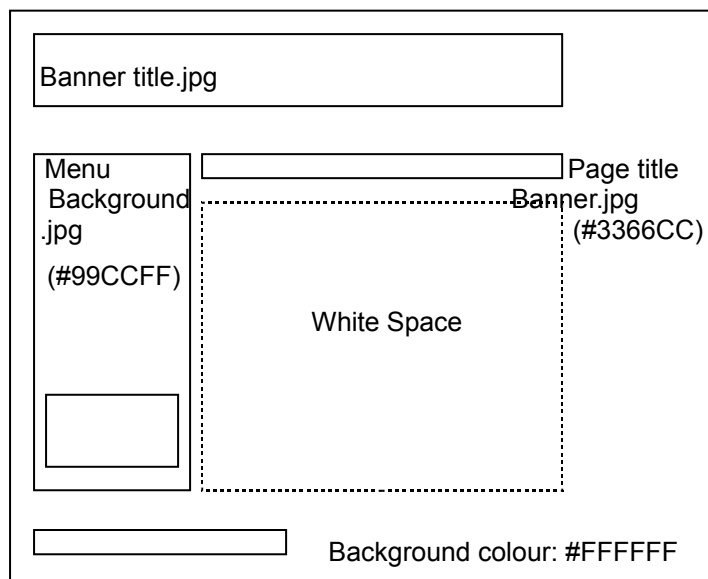


Fig 5.1

Appendix E contains screenshots of the main pages within the website (electronic version only)

Once a layout was finalised, image designs for banners were created using Microsoft Photodraw. Also the colours for the website had to be decided upon.



It was decided that the use of images would be kept to a minimum, as they would minimise the space needed for the required information.

Colour within the website will also be kept to a minimum in order to keep the interface recognisable throughout the website. Blue will be the focal colour as it's bold and will be used to represent significant areas of a web page such as page titles and the static links menu background. The dynamic area will be white apart from the aerial images section where a shade of blue will be used as a background. The background significantly highlights the aerial image that is placed on top.

Body text within the website will be black due to its strong contrast with white and blue. Text headings will be in blue. Bold yellow text will be included for page titles as it stands out well against the blue title bar.

The next step in the design stage was to produce a navigation chart that would be used within the implementation stage. A navigation chart's purpose is to outline how the pages of the website will be structured. The use-case diagrams mentioned within the analysis chapter aided the production of navigation charts as they state which pages will be necessary for various functions.

The navigation chart can be found in appendix F.

One key issue when it came to the design was positioning. The layout of the interface wasn't large enough to fill the full browser window. Hence, a decision on the position of the interface had to be made. It was decided that the website interface would be positioned to the left hand edge of the browser window as this will allow any screen resolution to be used whilst viewing the website, without affecting the positioning or structure of the data within the interface. The decision not to use nested layers and tables for the implementation of the website was due to the fact that it would have been difficult to incorporate my designs into this structure.

The final design is based around the user using a 1280 x 1024 screen resolution. Lower resolutions will work fine although the entire interface may not be visible within the browser window, forcing the user to use the scroll bar.

5.1.1 Design Of Aerial Images Section

To allow the user to view aerial images, they must first search for the town they wish to view. As mentioned in the analysis chapter, a questionnaire was produced for the purpose of collecting information on search functions that are use throughout the Internet. From the results of the questionnaires, the following two search functions were chosen:

Interactive Map

Text Search

Many versions of the interactive map were designed. The final version incorporates elements from the earlier designs. The map is designed to keep any information not needed by the user hidden. The map displays the five districts of Tees Valley:

- Redcar & Cleveland
- Middlesbrough
- Stockton-on-tees
- Hartlepool
- Darlington

When a district has been selected, the towns and villages from that area are then displayed to the user. Fig 5.2 is a screenshot of the interactive map.

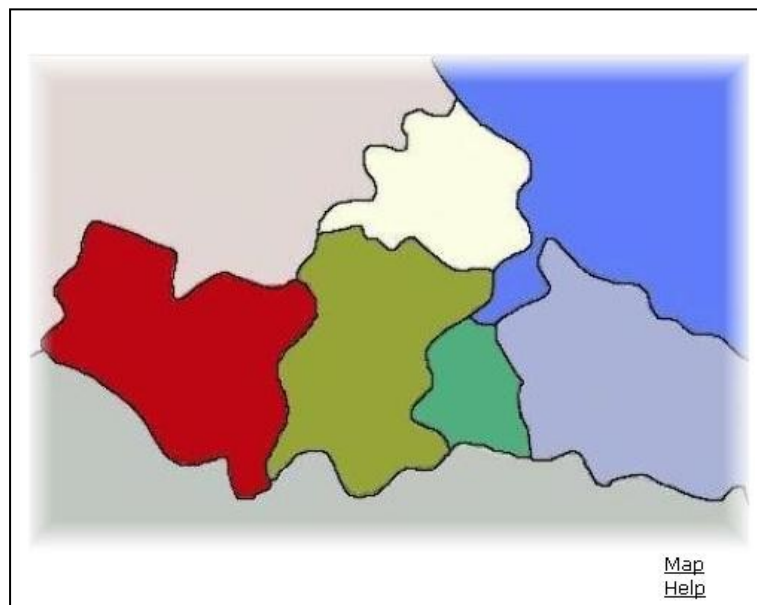


Fig 5.2

Different colours were used for the districts to allow the user to distinguish each area. By looking at the screen shot, there is no textual information displayed immediately. The district names are displayed to the user at the top left of the map when the user moves the mouse cursor over the appropriate area. By clicking on an area, the town names are displayed beneath the map area. A link to the help page is apparent so that users uncertain of how to use the map can view instructions.

The pages that display the aerial images are the focal point of the website. Within an aerial image page, the following functions must be displayed/accessible:

- An aerial image
- Navigation compass
- Overview map function
- Road map function
- Link back to views
- Link to history page
- Image Scale
- Link to help page

The design of the dynamic area of the aerial pages is shown in fig 5.3

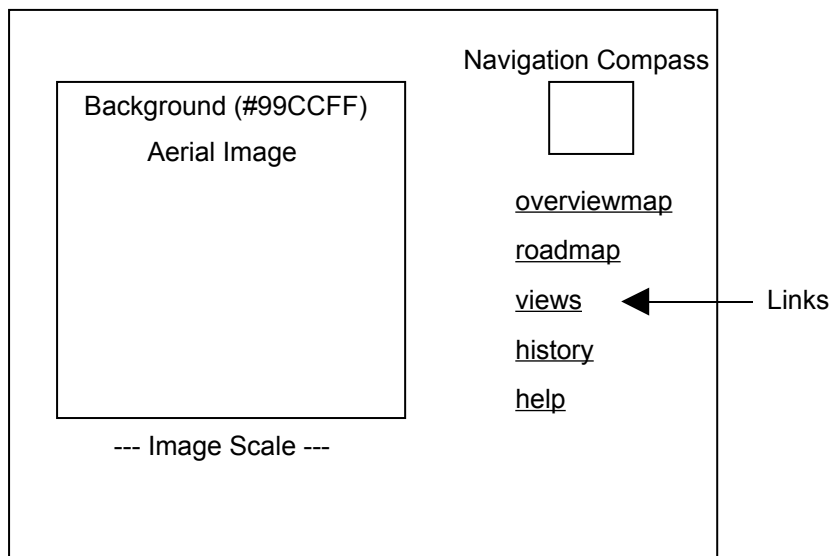


Fig 5.3

To allow for the overview and road maps to be displayed alongside the aerial images without having to expand the design, they will be placed inside popup windows. The links to these windows are found beside the aerial image. These windows when opened will appear at the top left hand corner of the browser window. The user will be able to reposition the window accordingly once opened.

The overview map is a single image of the entire town of Marske. The area of Marske that relates to the current aerial picture being shown is highlighted in blue. The user can then clarify their current position.

The road map pop up window contains an image that displays all the roads around the area of the current aerial image. The roads on the map that are displayed within that aerial image are displayed in a red highlighted area. Roads that surround the image are also shown outside of the red area. The reasoning behind this is that it shows the user where each road leads to, instead of just cutting off half of the road as this could cause confusion. Screen shots of both the overview map and road maps are found in Appendix E (electronic copy of report).

To allow the user to navigate around a town's aerial images in the least number of clicks, each individual aerial picture will cover a large area of ground. By using images that cover only a small area of ground, it would slow down the roaming process. It also cuts down on the amount of images that need to be produced. An example of this is shown in fig 5.4

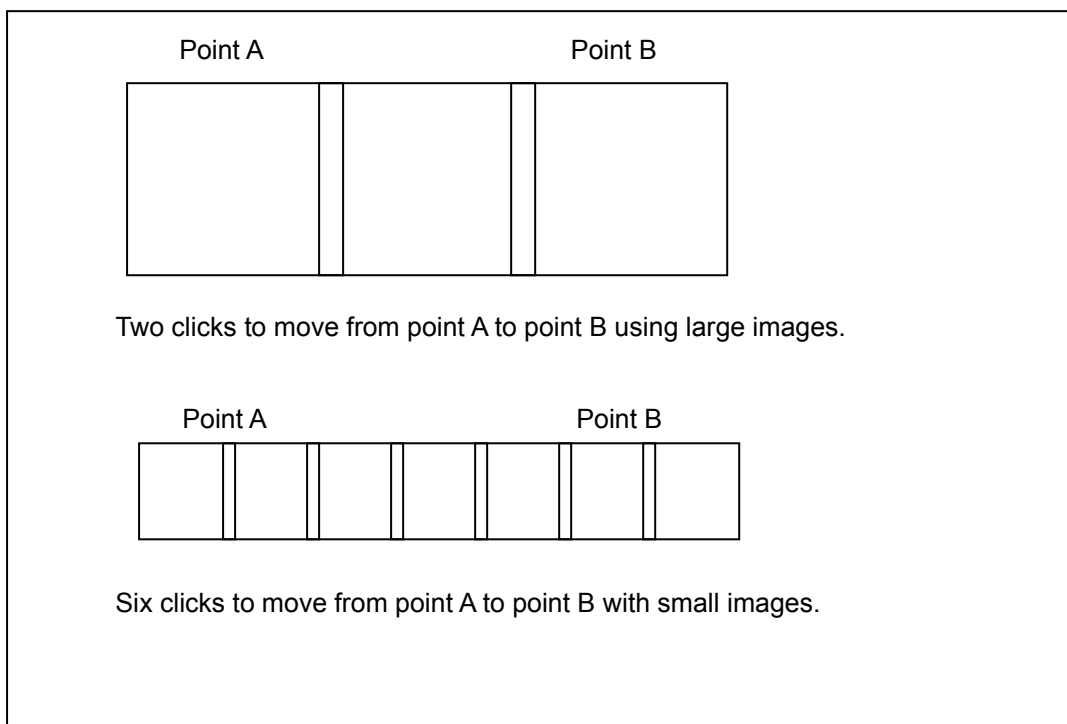


Fig 5.4

Each aerial image must be detailed enough for houses to be clearly visible. Hence, the images that will be downloaded from Multimap must be on a low scale. The chosen scale was 1:5000. Using a smaller scale would harm the image quality; a larger scale would take away the detail of the image.

5.1.2 Design Of Attractions Section

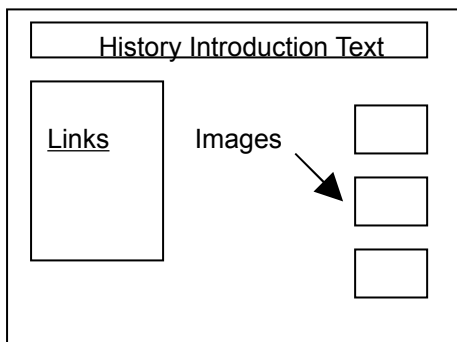
The design of the attractions section is very basic as only a form is required. The form consists of three text boxes and three associated buttons. These forms will be discussed further within the Chapter 6, which documents the implementation of the forms.

The user will enter the desired information into the required text box and click the search button. The information will then be displayed within a table.

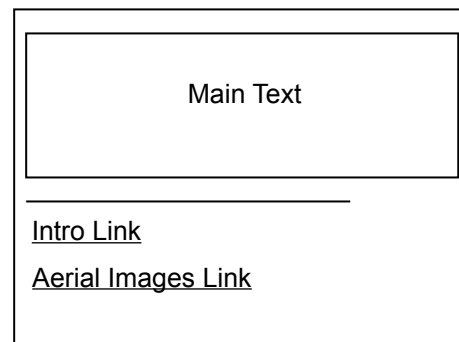
5.1.3 Design Of History Section

As there are no PHP functions within the history section, forms are not required. Hence, this section is populated only by text and images.

The introduction page will consist of a number of web links and some historic images to fill the white space in the dynamic area. Each link within this page directs the user to a second page that contains the historic information based on the town chosen.



Intro Screen



Town History Screen

In order to keep the amount of text on the web page to a minimum, at the end of each text section there is a link that takes the user a web page that contains extra historical information. As well as a link back to the history introduction page, there is also a link to the town's associated aerial images. This link appears only within the Marske page.

5.1.4 Design Of Administration Section

As the user will not access this section, only a basic design is needed. The main menu of the admin section contains three links that takes the administrator to the required function. The three functions within this section are:

Add Attraction to database

Delete Attraction from database

Update attractions database

These functions are carried out via the use of standard forms. With each form, a link to the current attractions table is displayed within a pop up window. This will allow the administrator to view the table information for confirmation of the data already existing.

5.1.5 Design Of The Help Page

The purpose of the help page is to guide the user through the available functions within the website. The FAQ's will be situated at the top of the dynamic area. Each FAQ is linked to the accompanied answer located below. This design allows the user to select the required information, which in turn directs the user to the answer. Thus, not giving the user the extra work of having to scroll through a large amount of text.

| |
|--|
| <p><u>FAQ's</u></p> <p><u>1.....</u></p> <p><u>2.....</u></p> <p>1. FAQ Title</p> <p>Related Answer</p> |
|--|

5.2 Design Of The Database

The database is in place to store information which will be displayed within the website. There are no data relations between the tables within the database as its not required as no functions within the website depend on information from more than one table.

The attributes of each table will be discussed below. The data dictionary contained in Appendix G defines the constraints of each table attribute.

The aerial images displayed on the website are linked to via the database. The table consists of an ID and links to the aerial images that is stored within the intranet. The ID number identifies each image. An example of the contents of the 'Images' table is shown below:

| ID | Link |
|----|---|
| M1 | http://wwwscm.tees.ac.uk:8002/users/a1021040/Website/Aerial%20Pages/Images/MARSKE/m%201.jpg |

The 'Views' table provides information to the website to allow the user to select which aerial image they wish to view first. For each view, a short description is given as well as a link to the aerial page in which that image is stored. Below is an example of a possible view record:

| Description | Link |
|-------------------------|--|
| Longbeck Trading Estate | /users/a1021040/Website/Aerial%20Pages /Marske/Marske%201.php |

The 'Attractions' table contains information concerning the places of interest in Tees Valley. Each record details the ID of the attraction, the type of attraction, the location, the name, and a link to the homepage of that attraction. An ID is given to each attraction for administrative purposes. A record held within this table would appear as follows:

| ID | Type | Town | Attraction | Website |
|-----|----------|--------|------------|-------------------|
| a21 | Outdoors | Redcar | Beach | http://www ... |

The 'places' table consists of just one data record. This table is used on conjunction with the aerial images search text box. Within the table, the town name and link to that town's aerial views is stored. In the case of more towns being covered by aerial photography, required records would be added to the table.

| Town | Link |
|--------|---------------|
| Marske | http://www... |

CHAPTER 6: IMPLEMENTATION AND TESTING

This chapter will discuss how the website and database was implemented using the Evolutionary method. Some sections will contain example code that relates to the discussion. Full source code is found in Appendix H. A brief discussion of the testing is documented at the end of this chapter.

6.1 Implementation Of The Evolutionary Versions

As mentioned in the methodology chapter, the implementation of the website was developed incrementally.

The web application end of the prototype was developed first, followed by the implementation of the database. The final stage was to link the web application to the database. The following sections will describe in order of versions, which pages and functions were implemented.

6.2 Version One

6.2.1 Creating Templates

The first step in creating the website was to develop a set of templates which would form the basis of the different web pages. The templates consisted of the required basic needed by each page such as quick links, web banner and page titles.

By creating the templates, it allowed me to create all the web pages needed in a short space of time. The implementation of a single webpage derived from a template required only the addition of the required dynamic data for that page. The layers for dynamic data were drawn so that when it came to adding the data, it was a simple matter of highlighting the layer and placing and required coding within the layer tags.

Once the full file structure of the website was in place using the templates, the next version could be implemented.

6.3 Version Two

6.3.1 Functionality Of Main Web Pages

Now that the website structure is in place, the main pages within the website that do not require any forms or use of PHP are constructed. These pages are:

- Home Page
- Aerial Image Search Page
- History Pages
- Help Page

During this period of implementation, the interactive map and aerial images were implemented.

The first page to be populated with data was the home page as this is the starting point for the user.

With the template in place, all that was required to create the rest of the page was to import the necessary images, buttons and text. Each button was then linked to the required pages.

The next web page to be implemented was the aerial images search page. To recap, the two search functions available to the user for the images search are:

- An interactive map
- A text box search

The text search requires the use of a form and PHP coding. These functions will not be implemented in this version of the prototype. Hence, I will only discuss the implementation of the interactive map at this stage.

The first step to implementing the map was to create an image of the Tees Valley area. The image was designed so that each district within Tees Valley was highlighted, allowing the user to browse each area separately.

Once the image was placed within the web page, a number of image maps drawn within the borders of each district area. Along side these image maps, a number of hidden layers were created, each layer displaying the name of each of the five districts. A '*mouse enter*' and

'*mouse leave*' function was then created for each image map. When the mouse enters the area, the district name appears until the mouse cursor leaves.

The map to this point went through a testing stage to ensure that the correct information was displayed when needed. To finalise the map, another selection of hidden layers were created underneath the map image. These layers contained a list of the towns within the related district, with each town name acting as a link to the aerial images. These town names were displayed when the user clicks on an image map. In the case of the prototype, Marske was the only town name to be converted to a URL link.

To ensure that a section of the website was fully functional before more detail functions were implemented, the history pages were the first set of pages to be fully produced.

The implementation of the pages was achieved in very little time. The necessary text was placed within the dynamic layer that was created as part of the template. Links were then created to allow the user to navigate around the history section.

The final page to be produced for the current version was the help page. The help page is in the form of frequently asked questions. The first stage when it came to producing the page was to create a list of FAQ's at the top of the dynamic area. A larger list was then included, consisting of the FAQ question and the answer located beneath.

Links were then included, to allow the user to jump from the FAQ's to the answer, and then back to the FAQ's.

6.3.2 Implementation Of Aerial Images

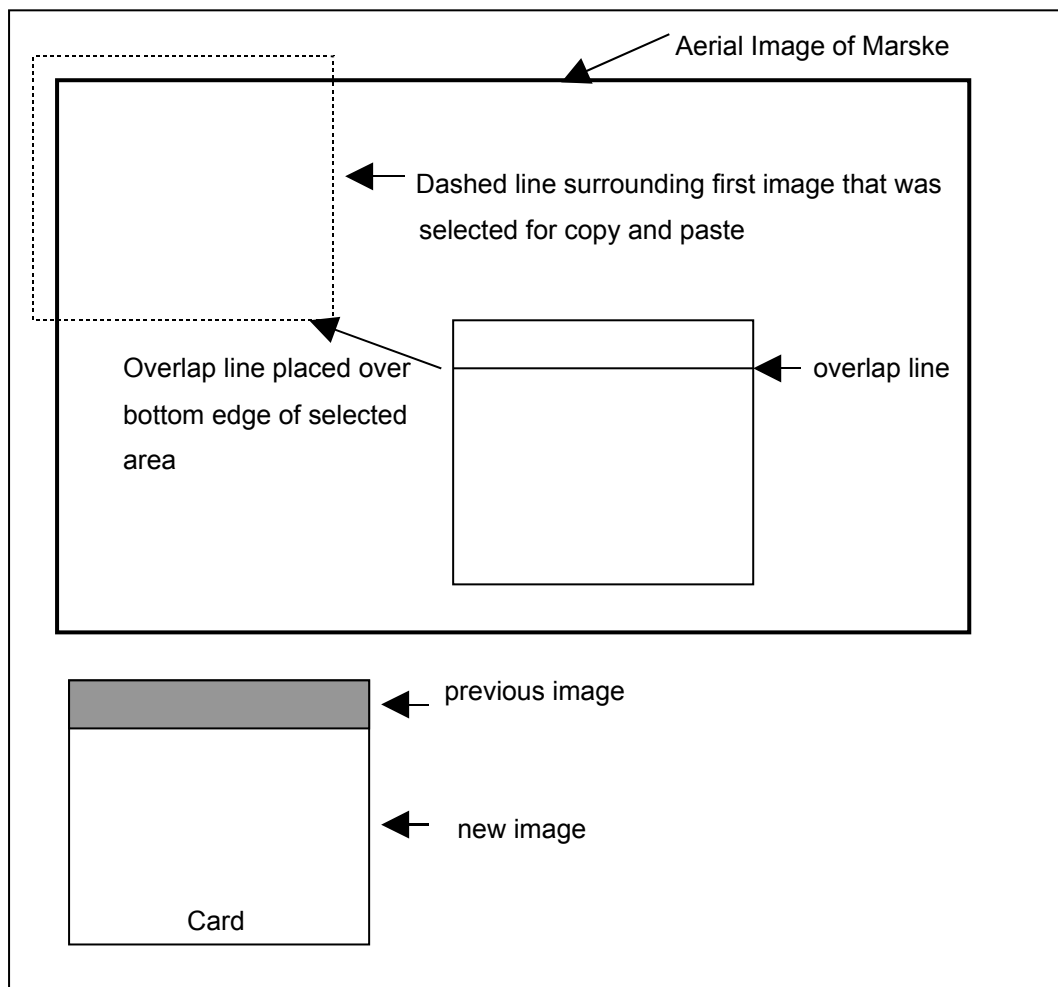
I created a method that would allow me to convert the images available online to the image format needed for my website. Before this method idea came to mind, I had tested software programs such as Photoshop and Microsoft PhotoDraw. I found that neither of these tools allowed me to do what I had in mind. The following is a discussion on my created method.

The first process was to download the required images from Multimap. The reason for downloading the images from Multimap is that each aerial picture is a single image. Get mapping images are made up of a number of smaller images; hence it would have taken longer to download the images. Each image was then joined together within PhotoDraw to make one large image of Marske. The large image was saved, ready for the following process, which was to split the image back into individual sections. Each individual image must be the correct size and also have an overlap. The method used to cut the image into equal sizes was very basic. It involved cutting out a square piece of card 4" x 4" in size with a line drawn down one side 1" from the edge. This line will act as a guide to the overlap of images.

The piece of card was placed over the top left hand side of the image of Marske, which was uploaded into Microsoft Paint. The select tool was used to draw around the outside of the card shape. The card was then removed from the screen, and the selected section of the image was then copied and pasted into a new paint window, and then saved.

Going back to the large image of Marske, the area that was selected for copy and paste is surrounded by a dashed line. In order to get an overlap between images, the card is placed beneath the dashed line with the overlap line on the card being level with the dashed line.

The diagram below is a graphical example of how overlapping was achieved.



This process of splitting was carried out until each individual image was saved. The next process was to resize each image so that they would fit into the design of the web page. The final stage was to use a tool in PhotoDraw called 'Sharpen' which cleared each image slightly and brought out more detail.

6.4 Version Three

6.4.1 Implementing Forms

A number of functions within the prototype website require the use of forms [12]. These forms will require data to be entered so that information can be retrieved from the database. The forms needed are:

- Aerial Image search box form
- Attractions search forms
- Administrator password form
- Admin update, add and delete attractions form
- Membership sign up form

One benefit of using Dreamweaver to implement the site is that the form can be implemented either by selecting the form objects from a drop down menu or by hand coding the HTML to create the forms. To speed up the implementation, the objects were selected via the menu. Each object was then named accordingly.

The image below shows the textfield and button object within a form. Below the image is the code that was created for the form.



The image shows a search form with the text "Search by Attraction Name:" followed by a text input field and a "Search" button.

```
<form onSubmit name="AttractionForm" method="post" action="attractiontable.php">  
  <input name="attraction" type="text" id="attractionfield" maxlength="200">  
  <input name="SubmitAttraction" type="submit" id="SubmitAttraction" value="Search">  
</form>
```

The user inputs information into the text box and by clicking the submit button, the user is then directed to a PHP page '*attractiontable.php*' from where the relevant information is given back to the user. This will be in evidence in later versions of the website.

The forms for updating, adding and deleting were linked to javascript validation files. This ensures that no text boxes can be left blank before a function is carried. If blank information is

sent to the database, this may cause functions not to work properly. To link a form to a '.js' file the following code is needed:

```
<form onSubmit="return validateForm(this)"
```

The following code is placed within the <head> tags of the same page as the form.

```
<script src="filenameEX.js"></script>
</script>
```

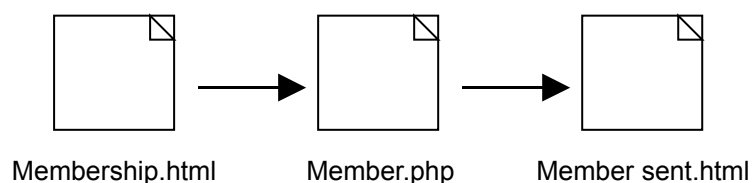
An example of the data within the validation forms is now shown:

```
function validateForm(data)
{
    var send=true;
    {
        data.ID.focus();
    }
    if (data.ID.value == "")
    {
        alert("Enter Attraction ID!");
        data.ID.focus();
        return false;
    }
}
```

The code shows that if the value inside the textbox named 'ID' is left blank, the alert function will be called.

6.4.2 Implementing the Membership section

The member section requires three pages, which are shown below:



The first page requires a form which was implemented in the previous version of the prototype website. This allows the user to enter their name, password and email address. Once the user clicks the submit button, the information within that form is sent to the second page. The page '*member.php*' holds the first PHP coding developed for the website [8]. This page handles the information sent from the first page. The information is stored within variables and then placed within some HTML script. The data is then sent to the email address supplied by the user.

```
<?php
// Information from forms placed into variables
$email=$_POST['email'];
$name=$_POST['name'];
$pass=$_POST['pass'];

// Message that's sent to email address

$message="Dear Customer, this is a message to confirm you have
applied for membership to our website.
You will be sent a following email within 24 hours with a link you
must visit to confirm to us that you do wish to be a member.

Your details are as follows:

Name: $name
Password: $pass
Email: $email

(please keep these details for security reasons)
Thank You,
Tees Valley Team";

if(mail($email,"Membership details",$message,"From: member-
services@TeesValley-air.co.uk")){
echo "Sending mail...";
} else {
echo "There was an error whilst sending the email, please make sure you
have filled in all fields correctly.";
}

// Directs use to member sent page

$url= "member sent.html";
```

```
header ("location:$url");
```

```
?>
```

The script '\$url= "member sent.html";' directs the user to the stated web page once the email has been sent. The member sent page displays a few lines of text that confirms the email has been sent. A link back to the first member page is also included.

6.5 Version Four

6.5.1 Implementing The Database

Using secure shell, the tables of the database were created and then populated with the required data.

A separate PHP file named '*connection.php*'; was developed. This file contains the information needed to connect to, and access the database, which all PHP pages need to do. Hence, if the information were to change, it can be edited within this file and it will update the connection function within each PHP file. This avoids having to change the details in every page.

To ensure the information from the database is displayed when it comes to creating the PHP code for the various functions within the website, a test page was created. The purpose of this page was to test the connection to the database [7] and to ensure that the data is correctly displayed within the table that will hold the information.

Note that all examples of coding found in the following sections highlight only the coding required to carry out the functions. Full source code is found within appendix H.

The code to create a connection to the database is as follows:

```
$connection = mysql_connect("$dbhost", "$dbusername", "$dbpassword");  
mysql_select_db ("$dbusername", $connection);
```

The variables '*\$dbhost*', '*\$dbusername*' and '*\$dbpassword*' are global and relate to the server that hosts the database and the username and password that are needed to access the database.

The first line connects MYSQL to the website. The second line selects which database will be accessed.

To select the data that is required, a query is written.

```
$query = "select InfoA, InfoB FROM tablename;
```

The above query selects information from the required table. If only one record of information is required from the website, a *WHERE*; statement is included after the table name.

The next stage was to create a table to store the information on the web page.

```
<table class="tablestyle" border="1" align="center">
<tr class=header>
<th class=styleth>InfoA</th>
<th class=styleth>InfoB</th>
</tr>
```

The above code creates the table headers and columns. The table class is linked to a local table style that manages the design of the table.

```
<?php
while ($row = mysql_fetch_array($info))
{
$InfoAvariable=$row["InfoA"];
$ InfoBvariable =$row["InfoB"];
}
```

The while loop counts each row of the table and places the information into a variable. This variable is placed into a row of the required column. The while continues until each row of the table has been read. The data is then echoed into the relevant table columns.

```
echo "<tr>";
echo"<td>";
echo $ InfoAvariable;
echo"</td>";
echo"</tr>";
}
?>
</table>
```

The code was tested until data was displayed. Once working, the code was copied and pasted into all PHP pages. The code was then edited accordingly.

6.6 Version Five

6.6.1 Allow The User To View Attractions

To recap, the search function within this section allows the user to view the information stored in the attractions table in three ways. These are:

By Attraction Type

By Town Name

By Attraction Name

The form and links to the PHP pages, which will display the attractions information, were already developed in an earlier version. Hence, to implement this section fully, only the PHP code needed to be included. The PHP code is the same format shown in the examples in the previous section. The only change to the coding was the query.

The query selects all the information from all records that contain the town name that matches that of the name entered by the user. Within the table 'attractions' there are URL's for the websites of the attractions. To display the URL as a link in which the user can click, an 'a href' tag is placed within the PHP code

Search.html

```
<form onSubmit name="Townform" method="post" action="towntable.php">
<input name="town" type="text" id="townfield" maxlength="25">
<input name="Submit" type="submit" id="SubmitTown" value="Search">
</form>
```

Towntable.php

```
$query = "select Type, Town, Attraction, Website FROM attractions WHERE
town='$town'";
```

This line shows how the URL tag is placed around the website variable.

```
echo"<td>";
echo "<a href=\"{\$atWebsite}\">Click Here</a>";
echo"</td>";
```

6.6.2 Allow User To Use Text Box Search For Aerial Images

This function works using the same method as searching for an attraction. The user inputs the name of a town they wish to view in the 'search.htm' page. A query then selects the information attributed to the name of the town and displays it within 'search2.php'.

6.6.3 Display Aerial Views

The PHP coding required to display the views uses the same structure as the code to display the attractions data. The only different being there is no 'WHERE' statement within the query as its not required. The PHP code used for displaying attractions was copied and pasted into 'views.php' and the query and variables edited accordingly.

6.7 Version Six

6.7.1 Display Aerial images and Pop up windows

The URL of the aerial images that are located in the intranet space is stored within the database. When displaying the URL using PHP, an 'img src' tag is used so that the image is displayed instead of the URL of the image.

The query is written so that only the image that is needed is selected and displayed. Each image is given an ID, for example the aerial image to be displayed within 'Marske 1' is given the ID 'm 1'.

```
<?php
$query = "select Image FROM aerialImgs WHERE ID='m1'";
$info = mysql_query($query,$connection);

$Aimage=$row["Image"];

// placing image variable in image tag
echo "<img src=$Aimage>";
}
?>
```

The PHP coding is placed within the <div> tag of the layer that will contain the image. No table is required to display the image.

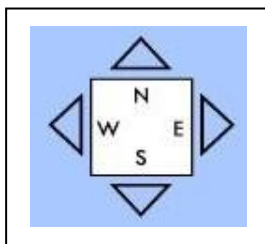
For the overview and road map functions, popup windows have to be implemented using JavaScript. The script is placed within the <head></head> tags of the page.

```
<script>
<!--
function openMap(){
var popurl="MAP/Map 1.htm"
winpops=window.open(popurl,"","width=600,height=470,scrollbars,top=30,left=
45")
}
//-->
</script>
```

The link to open the road map is linked to the function name of the Javascript. The '*popurl*' line states which page will be displayed within the window. The details contained within the brackets control the size and position of the pop up window when displayed.

6.7.2 Implement Navigation Compass and links

The navigation compass comprises of four compass points and a compass face. Each point is an individual image that is linked to the page that it should lead to when clicked. The image below is a screen shot of the compass.



The links to direct the user to the help page, Markse history and back to the aerial views page were the last parts to be implemented in this version.

6.8 Version Seven

6.8.1 Allow Administrator To Edit Attractions Table

In the case of this prototype, only the attractions and password table within the database can be fully updated to show how the functions will operate. The other tables stored within the database contain information that is unlikely to change. Hence, it was decided not to include functions to update that information.

6.8.2 Adding Attraction

To add an attraction, the admin must enter the attractions information in the form located at '*admin add.htm*'. Radio buttons are used to select the attraction type as this ensures that the correct information is added to the database. Text boxes are used for the rest of the input as unlike the attractions type, the rest of the data is dynamic. The information is then sent to '*admin add.php*', where the query to post the information to the database is located. The first section of code below shows only the attraction section of the form.

Admin add.htm

```
<form onSubmit="return validateForm(this)" name="addform" method="post"
action="admin add.php" >
  <table>
<td>Attraction:</td> <td><input name="attraction"
type="text" id="attraction"></td>
  </table>
  <input name="Add" type="submit" value="Add">
</form>
```

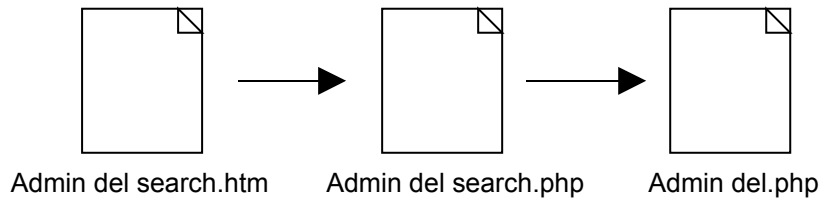
Admin Add.php

```
$query = "INSERT INTO attractions(ID, type, town, attraction, website)
VALUES ('".$_POST['ID']."', '".$_POST['type']."', '".$_POST['town']."', '".$_
$_POST['attraction']."', '".$_POST['website']."'");
```

The '*post*' function sends the information within the variable located inside the brackets '*[variable]*' to the matching attribute within the database table.

6.8.3 Deleting Attraction

The next function to implement was to delete an attraction from the database. This function was carried out using the same structure as the membership section, although the coding is different.



The ID of the attraction, which is to be deleted, is input by the administrator in '*admin del search.htm*'. The variable containing the ID is sent to the second page '*admin del search.php*'.

The '*admin del search.php*' page displays the ID, attraction name and location to the admin and two buttons 'yes' and 'no'. By clicking the 'no' button, the admin is directed back to the previous page. If 'yes' is clicked, the attraction information is sent to '*admin del.php*', which handles the query that will delete the attraction from the database.

Admin del search.php

```
$query ="SELECT town, attraction FROM attractions WHERE  
attraction='$attraction'";
```

```
// form for yes and no buttons
```

```
<form onSubmit name="YesForm" method="post" action="admin del.php">  
<input type="hidden" size="100" name="attraction" id="attraction"  
value="<?php echo "$attraction"; ?> ">  
<input name="Yes" type="submit" id="Yes" value="Yes">  
</form></td>
```

```
<td width="45%"><form onSubmit name="NoForm" action="admin del search.htm">  
<input name="No" type="submit" id="No" value="No">  
</form></td>
```

Admin del.php

```
$query ="DELETE FROM attractions WHERE attraction='$attraction'";
```

6.8.4 Updating Attraction

The final function for the administration section to be implemented is that of updating an existing attraction within the database. Using the same method as the *'delete'* function, the admin enters the attraction ID. The related attraction details are then displayed to the user in the form of a table. The details are also placed into text boxes to allow the information to be edited.

Admin Update Search.php

```
<form onSubmit name="addform" method="post" action="admin update.php" >
  <table width="75%" border="0">
    <tr>
      <td>ID:</td>
      <td><input name="ID2" type="hidden" id="ID2" size="5" maxlength="4"
value="<?php echo "$atID"; ?>"></td>
    </tr><tr>
      <td>Type:</td>
      <td>input name="type" type="text" id="type" value="<?php echo
"$atType"; ?>" size="20"></td>
    </tr><tr>
      <td>Town:</td>
      <td><input name="town" type="text" id="town" value="<?php echo
"$atTown"; ?>" size="20">
      </td>
    </tr><tr>
      <td>Attraction:</td>
      <td><input name="attraction" type="text" id="attraction"
value="<?php echo "$atAttraction"; ?>" size="40"
      ></td>
    </tr><tr>
      <td>Website:</td>
      <td><input name="website" type="text" id="website" value="<?php
echo "$atWebsite"; ?>" size="40">
      </td>
    </tr>
  </table>
  <p>
    <input name="Update" type="submit" id="Update" value="Update">
  </p>
</form>
```

The highlighted area of the code shows that the data held in variable *'ID'* is placed into a new variable named *'ID2'*. The ID textbox is hidden so that the ID cannot be edited. Two queries are used to update the table. The first query deletes the unedited information that was linked

to 'ID2'. The following query then inserts the edited information into the database. Technically, the queries used are the same as those for adding and deleting, just in this case they are used together to create an update function.

```
Admin update.php
<?php
$db = mysql_connect("hardcase","a1021040","wLLotx37");
mysql_select_db ("a1021040");

//a query, which deletes the old data from attractions table
$query = "DELETE FROM attractions WHERE attraction='$attraction2'";

//executes the query
$info = mysql_query($query);

// a query that updates updates the database with the new information
$query = "INSERT INTO attractions(type, town, attraction, website)
VALUES ('".$_POST['type']. "','"._POST['town']. "','"._POST['attraction2']. "'
','".$_POST['website']. "')";
$result = mysql_query($query);
?>
```

An *INSERT* statement is another method of updating information within the database. The insert method was tested but it did not function. Hence, the above PHP coding was implemented in its place.

6.9 Version Eight

6.9.1 Allow Administrator To Log In

The initial idea for the log in function was to use the same method of that used when entered a town name within the aerial images search page. If the password entered were correct, a confirmation message and link to the main menu would be sent back. For some unknown reason, the link would only guide me to the index page of the admin folder. A number of different links were tested, neither of which worked.

It was then decided that JavaScript [8] would be used for the login function although it's not the most secure method. A prompt box appears on a blank page requesting the admin user number. If the correct user number is input the admin main menu will load, else an error page will be displayed. This page contains a link back to the login page.

```
<SCRIPT language="JavaScript">
<!--hide
var password=prompt('Enter the password:', '');
var mypassword="password";
if (password==mypassword)
{
    window.location="admin main menu";
}
else
{
    window.location="password error page";
}
//-->
</SCRIPT>
```

JavaScript coding for the password function.

6.10 Testing Of The Prototype Website

The testing of the website is carried out to ensure that the entire prototype works and that each requirement is met. A set of standard tests was devised to check that all buttons and forms work and produce the expected response. Each set of tests are based on the eight versions of the prototype website that were produced. In each test case, the response is recorded and compared to the ideal response. Appendix I contains the records from the testing.

CHAPTER 7: CONCLUSION

During the analysis stage of the project, a set number of requirements were stated. From what has been implemented into the prototype website, each requirement has been achieved at different levels.

It was immediately apparent during the planning stage that the website would be a prototype as to produce a fully implemented website was not possible within project timescale. I originally wanted to cover four towns with aerial images but after many discussions with my tutor, it was decided that reproducing the same work would not benefit the prototype. It was at this point that I decided that only Marske would be covered with aerial images. The ideas for functions such as road maps and an overview map began to develop at the same stage.

The method that I created to convert the aerial images is still the best method that I had found for that process. Finalising the method did take a while due to testing of the processes. Once finalised, I found the method to be very simple to use, although repetitive. If I were to convert more aerial images in the future, I would still use the same method, although small changes could be made.

Looking back at the methodologies I could have employed for the development of my website, such as the John December method, I feel as though I wouldn't have benefited by using them. As I had my own set ideas from the very beginning of the project, being tied down by a particular method or processes may have hindered the progression of the project.

During the research period, much time was spent looking at the finer details of the reviewed websites. It wasn't until I came to the analysis stage that I decided that much of the information I had collected was irrelevant. Although, by studying the sites in depth, I discovered that many of my original ideas for the project could not be achieved, due to access to the necessary technologies.

Comparing the project timetable that I specified in the project specification to the actual time spent on each section of development, there are big differences. The implementation took a lot longer than the anticipated six weeks. There are many reasons for this such as coding errors and design issues.

If I were to implement the prototype website again, I would use the same methods and technologies as the final result has shown that the tools have allowed me to achieve the implementation of the prototype website.

Looking back, there were moments when I had spent time producing documentation and functions that in the end became irrelevant. If I could use that time again, I would have looked into the possibility of expanding sections of the website to accommodate some extra features. These features are discussed in the following chapter.

CHAPTER 8: RECOMMENDATIONS

There are a number of future enhancements that can be recommended and that would be necessary if the website were to be fully implemented:

1. To implement a full version of the website, it would have to support all aerial images of towns within the Tees Valley area.
2. The navigation compass would be redesigned so that diagonal movement is supported.
3. The administration section should be expanded to allow all elements of the database to be updated accordingly.
4. Validation would be added so that duplicate data cannot be entered into the database.
5. A member login function. Once logged in, the user would be able to use any extra functions that are referenced within the member page of the prototype.
6. The '*Attractions*' table could be extended to allow a rating of the attraction to be included. The user could rate the attraction and the results are then stored within the table. An extra search functions could then be implemented which would allow the user to search for attractions according to their rating.

