

Lincoln Cathedral Gallery

Designing for Dynamic Content - WEB2002

James Willock [james.willock@gmail.com]

11/19/2008

This report will lay out the planning, design and development processes which take place in the creation of a dynamic website. I will develop a dynamic gallery website for photographs of Lincoln Cathedral using technologies such as XHTML, CSS, Javascript, PHP and MySQL.

Contents

1. Introduction	4
1.1. Dynamic Content Overview	4
1.2. Proposal	5
2. Design Process Database Structure	5
2.1. Database Structure	5
2.2. Data Types.....	5
2.2.1. tinyint(<i>n</i>).....	5
2.2.2. varchar(<i>n</i>).....	5
2.2.3. timestamp.....	6
2.3 Database Security	6
2.4. Table Definitions	7
2.4.1. MySQL Table "lc_photos"	7
2.4.2. MySQL Table "lc_names"	7
2.4.3 MySQL Table "lc_admins"	7
3. Implementing the Database Structure and Data	8
3.1. Creating Tables.....	8
3.1.1. Creating "lc_photos"	8
3.1.2. Creating "lc_names"	8
3.2. Inserting Data.....	10
3.2.1. Data submission diagram.....	10
4. Initial Testing of Database.....	11
4.1. Retrieving Data.....	11
4.1.1. Retrieving Data Script Example.....	11
5. Dynamic Web Site Design	11
5.1. File Structure	12
5.2. User Interface Design.....	13
5.2.1. Final Front Page Design.....	14
5.2.2. Final Detail Entry View	15
5.3. Combining xHTML & CSS with PHP & MySQL	16
5.4. Browser Testing	17
5.4.1. Safari 3.2 (OS X 10.5.5).....	17
5.4.2. Camino 1.6.5 (OS X 10.5.5)	18
5.4.3. Firefox 3.0.4 (Windows Vista).....	19

5.5. Testing on Various Connection Speeds.....	20
6. Conclusion	20
7. Recommendations	21
7.1. Improvements.....	21
7.2. Proposal for a Different Dynamic Web Based Gallery	21
7.3. Considering alternative programming language	22
8. Bibliography / References.....	22
8.1. Code references and libraries	22
8.2. Website Testing	22
9. Appendices.....	22
9.1. Research.....	22
9.2. Code Listing.....	24

1. Introduction

1.1. Dynamic Content Overview

A dynamic website is one which fetches content from a source separate from its client side code, such as a database. This allows the separation of content from the websites codebase, allowing users with little or no experience with web technologies the ability to perform updates or maintenance on a website. In this case, this is achieved by leveraging the relational database technology MySQL as storage for website content, common client side technologies such as xHTML, CSS and Javascript to display the website, and PHP as an abstraction layer in between MySQL and the client side code.

A good example of a dynamic site is the BBC News website¹ which is required to store and keep up-to-date thousands of news stories and articles, as well as post breaking news in a variety of formats as quickly as possible. The BBC News site being dynamic makes this all possible – without the time-saving measures available, the BBC would be much slower in breaking news on its site than its competitors. Additionally, by separating design and content, the visual and structural makeup of the site can be changed and updated without needing to update thousands of pages as would be required on a static website.

In this case, I will be using two popular technologies, PHP and MySQL to form the server-side of my project, and deploying these on a Linux-based Apache webserver. I will develop the client and server side code on Mac OS X 10.5.5 using Panic's web development application *Coda*². Additionally, I will create and maintain the MySQL database and tables using the open source phpMyAdmin³.

PHP, or PHP: Hypertext Pre-processor, is a scripting language used on over 20 million domains⁴ used to create dynamic websites when embedded into HTML or other client-side languages. Its popularity is partly due to its open source nature (released under the PHP License⁵), it being weakly typed (allowing free conversion and movement between datatypes) and the relative ease in which someone new to scripting can become familiar with its workings. It also benefits from a large developer community which can provide support, suggestions and code libraries.

MySQL is a relational database technology based on IBM's SQL or Structured Query Language. MySQL allows the storage of large amounts of data in relational tables, which can be accessed using queries on the command line – or, in this case, using PHP. MySQL is popular for similar reasons as PHP – it is open source (licensed under GNU 2.0⁶), it has a large development community, and like PHP is cross platform.

¹ BBC News. <http://news.bbc.co.uk/>

² Panic. <http://www.panic.com/coda/>

³ The phpMyAdmin Project. <http://www.phpmyadmin.net/>

⁴ The PHP Group. <http://www.php.net/usage.php>

⁵ The PHP Group. http://www.php.net/license/3_01.txt

⁶ The GNU Operating System. <http://www.gnu.org/licenses/old-licenses/gpl-2.0.txt>

1.2. Proposal

I will plan, design and develop a dynamic gallery website, storing and displaying thematic artwork for viewing by my target audience. I will employ skills I have learnt over the course of my studies and previous, employing knowledge of information architecture, usability and accessibility.

The site will display thematic photographs of Lincoln Cathedral, taken by myself and users of the website. Additionally, the site will be designed to reflect the sorts of photographs that may be uploaded onto the website – I will use natural colours and design elements that will not clash with the displayed artwork. I propose the following features for the gallery website:

1. Home page listing several rows of the most recently uploaded images.
2. Pagination system for viewing pages of uploaded images.
3. “Detail” page for each image (dynamically generated), with fold out lightbox for high-resolution images.
4. Upload section allowing users to add their own images to the gallery.
5. A password-protected section allowing administrators to vet images, and accept or deny them.
6. Several information pages explaining the purpose and workings of the site.

2. Design Process Database Structure

2.1. Database Structure

The relational database for the dynamic gallery will contain two tables, “lc_photos” and “lc_names”. The former will store the gallery images and data attached to them (description, data uploaded, etc), and the latter will store the names of each image author. I will store the user data separately, since a user may upload more than one image, and storing that data more than once would be redundant.

Additionally, there will be a third non-relational table, “lc_admins” which will store the user details of the site administrators, used to control the content management system.

2.2. Data Types

Each column in a table must have a data type to define what sort of data can be stored within. The following will be used in the gallery database:

2.2.1. tinyint(n)⁷

The tinyint data type can store integers up to 1 byte (up to 255 different values). This will be used on the primary key ‘id’ column, and ‘processed’ column (where it will be given the parameter “1” and store boolean data).

2.2.2. varchar(n)⁷

The varchar data type can store string data up to 255 characters in length. It will be used to store short text data such as names and URLs.

⁷ MySQL Documentation. <http://dev.mysql.com/doc/refman/4.1/en/numeric-types.html>

2.2.3. timestamp⁸

The timestamp data type can store dates in a timestamp format (e.g. “2008-11-20 19:11:12”), which can be set to enter the current timestamp as default.

2.3 Database Security

Since the table “lc_admins” will store unique sensitive information about users – specifically their passwords, which they may inadvertently share with many other web services, it is good practice to secure this data in some fashion. In some cases, it may be possible for unscrupulous web users to force a web site to dump the contents of an SQL database or table into the browser or other application – if passwords and other sensitive data are not secured somehow, the user could simply search through the data and perhaps use it unlawfully.

In this case, it is good practice to use “hashing” in order to store passwords in a database. This is the process of taking originally submitted user data and applying a secure mathematical algorithm (in this case, using the PHP function sha1()⁹) to it in order to disguise its content. The “hashed” password is then stored in the database, and due to its one-way encoding, it cannot be reverse engineered without the use of “rainbow tables”. When details need to be confirmed, such as the password during the log in process, user entered data is “hashed” again and compared to the value stored in the database – if the two hashed values match, the password is the same and log in can be completed.

However, rainbow tables, which are vast collections of hashes that are known to associate with certain values, are readily available on the Internet. An unscrupulous user could take the hashes stored in your database, use rainbow tables to search for matching hashes and discover the un-hashed values. To prevent this, we can use “salt”, which is a random string attached to the un-hashed value before it has the algorithm executed on it – the salt is then also stored in the database and used when comparing user-submitted passwords to database-stored ones. This almost completely nullifies the dangers of rainbow tables.

⁸ MySQL Documentation. <http://dev.mysql.com/doc/refman/4.1/en/string-types.html>

⁹ The PHP Group. <http://uk.php.net/sha1>

2.4. Table Definitions

2.4.1. MySQL Table “lc_photos”

The table “lc_photos” stores the primary data of each entry, such as the image name, description, URL to image and the date uploaded.

Field	id	image_name	name	description	image_url	processed	timestamp
Type	tinyint(10)	varchar(255)	tinyint(10)	varchar(255)	varchar(255)	tinyint(1)	timestamp
Modifiers	not null, auto increment, primary key	not null	not null	null	not null	not null, default '0'	default 'CURRENT_TIMESTAMP'
Example	1	Tennyson Statue	1	A statue of...	1227208270tns.jpg	1	2008-11-20 19:11:12

2.4.2. MySQL Table “lc_names”

The table “lc_names” stores the name of each image author, and the column “lc_names.names_id” will be linked to the column “lc_photos.name” in SQL queries.

Field	names_id	names_name
Type	tinyint(10)	varchar(255)
Modifiers	not null, auto increment, primary key	not null
Example	1	jwillock

2.4.3 MySQL Table “lc_admins”

The table “lc_admins” stores the user details of administrators. The “admins_password” column stores the password of the user in SHA hashed format.

Field	admins_id	admins_name	admins_password	admins_email	active
Type	tinyint(10)	varchar(255)	varchar(255)	varchar(255)	tinyint(1)
Modifiers	not null, auto increment, primary key	not null	not null	null	default '0'
Example	2	RandomAdmin	32fd sdf459569fs922	random@random.com	0

3. Implementing the Database Structure and Data

3.1. Creating Tables

The tables having been designed and laid out, they can now be created using PHP or the command line interface. The following SQL queries can be executed on the command line, entered into phpMyAdmin, or entered into a PHP file which will execute the code on runtime. It is always good practice to test and verify your tables and SQL in a tool such as phpMyAdmin.

3.1.1. Creating “lc_photos”

```
CREATE TABLE IF NOT EXISTS `lc_photos` (  
  `id` tinyint(10) NOT NULL auto_increment,  
  `image_name` varchar(255) NOT NULL default '',  
  `name` tinyint(10) NOT NULL default '0',  
  `description` varchar(255) default NULL,  
  `image_url` varchar(255) NOT NULL default '',  
  `processed` tinyint(1) NOT NULL default '0',  
  `timestamp` timestamp NOT NULL default CURRENT_TIMESTAMP,  
  PRIMARY KEY (`id`)  
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;
```

3.1.2. Creating “lc_names”

```
CREATE TABLE IF NOT EXISTS `lc_names` (  
  `names_id` tinyint(10) NOT NULL auto_increment,  
  `names_name` varchar(255) NOT NULL default '',  
  PRIMARY KEY (`names_id`)  
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;
```

3.1.3. Creating "lc_admins"

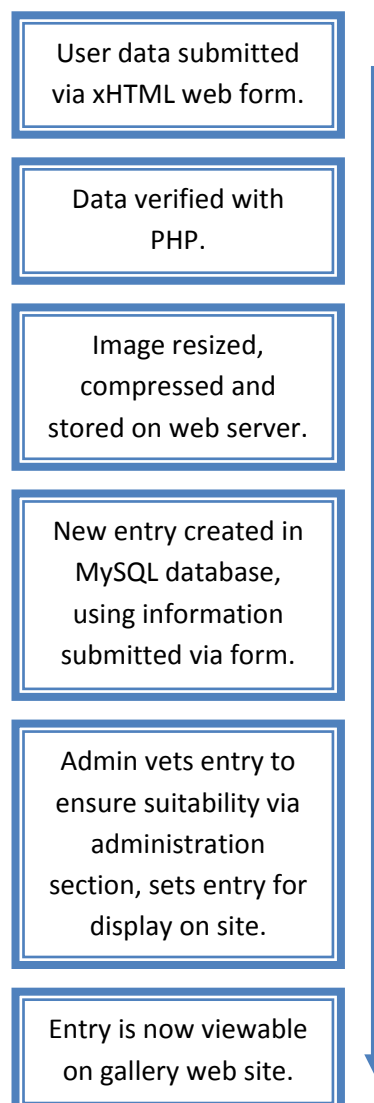
```
CREATE TABLE IF NOT EXISTS `lc_admins` (  
  `admins_id` tinyint(10) NOT NULL auto_increment,  
  `admins_name` varchar(255) NOT NULL default '',  
  `admins_password` varchar(255) NOT NULL default '',  
  `admins_email` varchar(255) default NULL,  
  `active` tinyint(1) NOT NULL default '0',  
  PRIMARY KEY (`admins_id`)  
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=3 ;
```

3.2. Inserting Data

In order for dynamic data to be displayed on a site via PHP and xHTML, it must first be entered into the database. This can be done manually, via the command line interface, through a maintenance tool such as phpMyAdmin, or through PHP, either by manually entering the data into MySQL queries, or in this case – creating a web form to take data from a user at the front end, validate it, and submit it to the database.

3.2.1. Data submission diagram

The process of a user submitting data, and it being displayed on site is shown in the following diagram:



4. Initial Testing of Database

4.1. Retrieving Data

In order to ensure that the data entry workflow has been successful, the database should be checked. This can be achieved, once again, via the command line, a tool such as phpMyAdmin, or via a PHP script written to fetch data from the aforementioned tables.

4.1.1. Retrieving Data Script Example

The following script could be uploaded to the dynamic site and run to check for data within the tables, and confirm data entry success.

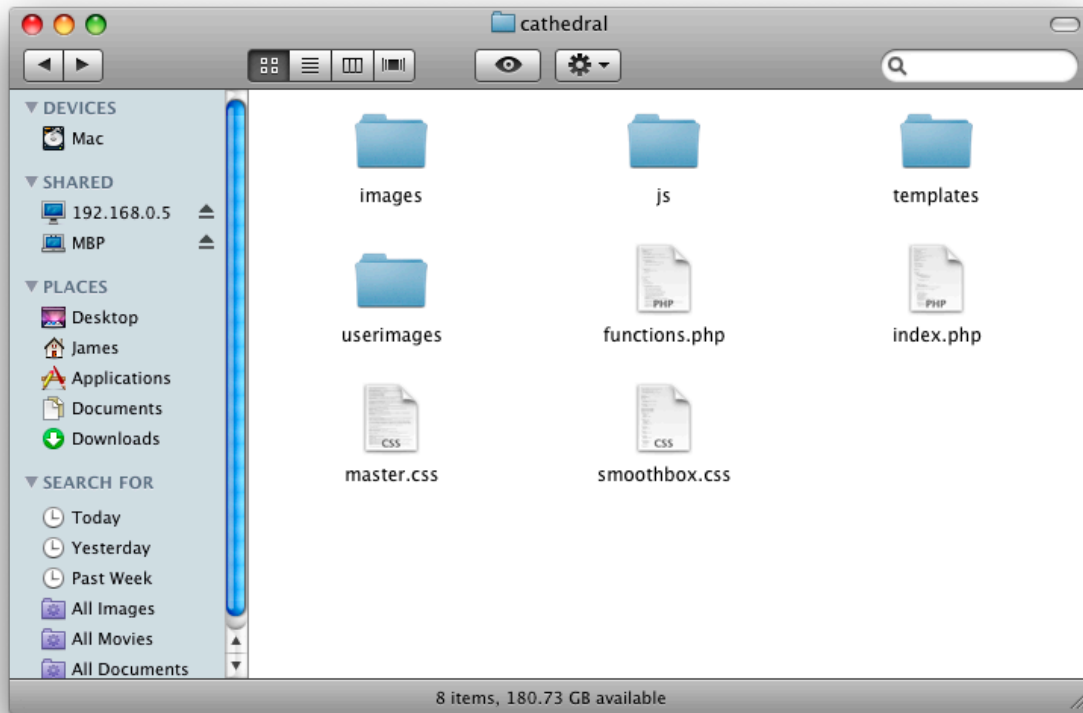
```
<?php
mysql_connect("localhost", "exampleuser", "examplepassword");
mysql_select_db("dyn_jwillock");
$result = mysql_query("SELECT * FROM lc_photos LEFT JOIN
lc_names ON lc_photos.name = lc_names.names_id");
while ($row = mysql_fetch_object($result)) {
    echo $row->id;
    echo $row->image_name;
    echo $row->names_name;
    // echo further rows
}
mysql_free_result($result);
?>
```

5. Dynamic Web Site Design

The code for this project has all been written specifically for this web site, with the exception of a bespoke templating class which has been used on my NewMedia portfolio. The PHP written for this site is listed in the Code appendices.

5.1. File Structure

The site's files are laid out in the following structure:



- “index.php” stores the main site procedure.
- “functions.php” stores the class, object and method library. It stores the templating, image reformatting and image upload classes.
- “master.css” is the site’s primary stylesheet.
- “smoothbox.css” is an accompanying stylesheet for the Javascript “Smoothbox” which allows pop-in high resolution images to display on screen.
- “/images/” stores the site’s graphic files.
- “/js/” stores the site’s DOM scripts.
- “/templates/” stores the site’s template files (*.tpl files, which store the site’s HTML and are called via the template class’ methods).
- “/userimages/” stores the user uploaded images, including high resolution and thumbnails respectively in the subfolders “/highres/” and “/thumbnails/”.

5.2. User Interface Design

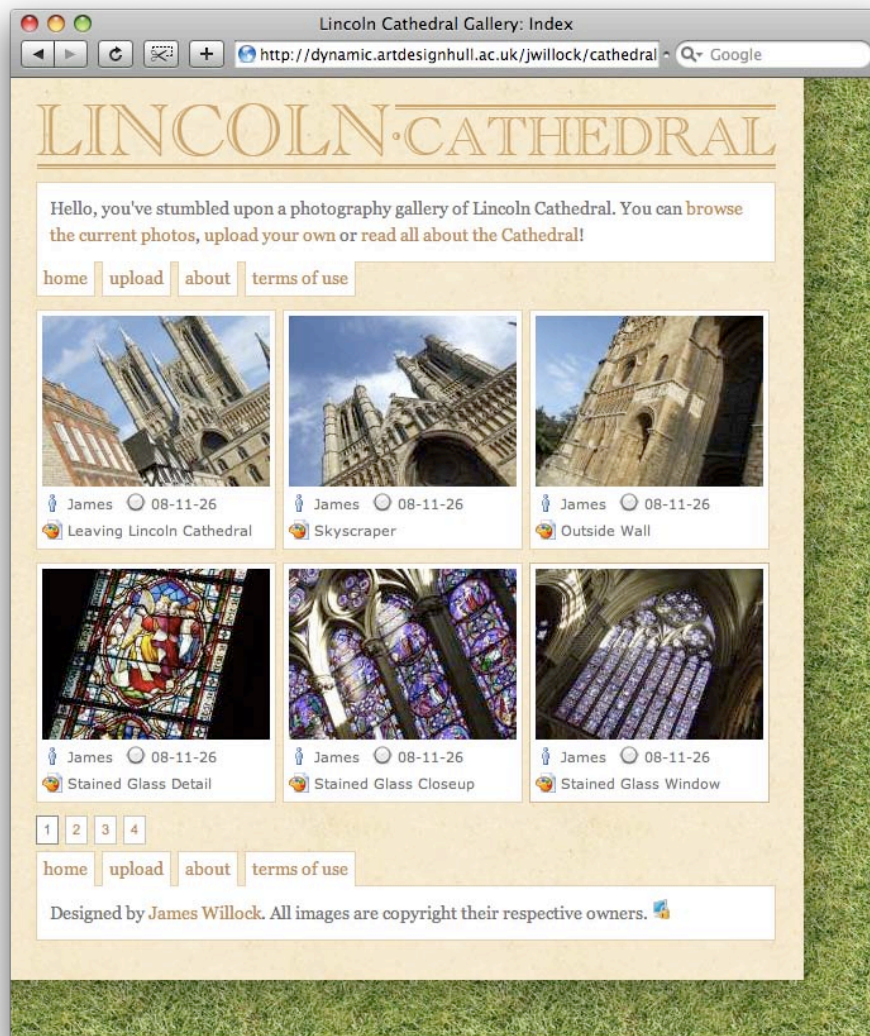
Developing a good user interface for the gallery website is an important aspect of the project. The site's visuals must complement the uploaded images in order for the site to be considered thematic, but the visuals must not detract from the images which are the primary content of the web site.

I determined that the site must have the following before developing any visual ideas:

- The front end must be coded in standards-compliant xHTML 1.0 and CSS, and viewable in modern web browsers.
- The site must be developed using sensible usability guidelines. It must be easily usable by anyone with web experience, and easily recognisable as a "gallery website" on the home page.
- The site must be developed with usability in mind, and not create problems for lesser abled users.
- The ability to upload an image of Lincoln Cathedral to the site, without the requirement to "sign up", using an attractive, usable web form. The form must have both client-side and server-side validation to prevent misuse.
- High-resolution images will be displayed using a Javascript "Lightbox"¹⁰ to display the image within the browser window at the highest resolution available. This means the main site does not need to be constrained to usual dimensions, and may be smaller in width than a usual gallery web site.

¹⁰ Wikipedia. [http://en.wikipedia.org/wiki/Lightbox_\(JavaScript\)](http://en.wikipedia.org/wiki/Lightbox_(JavaScript))

5.2.1. Final Front Page Design



5.2.2. Final Detail Entry View


Lincoln Cathedral Gallery: Image

http://dynamic.artdesignhull.ac.uk/jwillock/cathedr - Google

LINCOLN CATHEDRAL

You're looking at an image in the gallery. **Click the photo for a pop-out view!**

[home](#) [upload](#) [about](#) [terms of use](#)



An image showing the Cathedral in the background. In shot, the Tudor building housing the Tourist Information office.

Leaving Lincoln Cathedral
By: **James**
Uploaded on: **08-11-26**

← Skyscraper

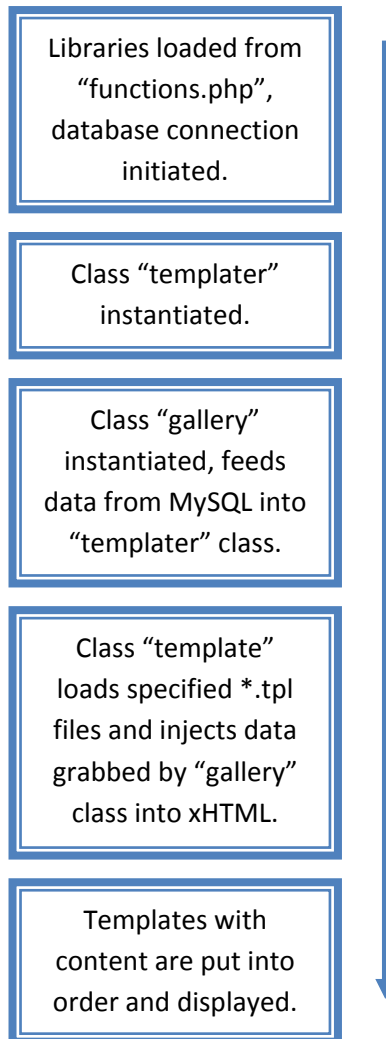
[home](#) [upload](#) [about](#) [terms of use](#)

Designed by **James Willock**. All images are copyright their respective owners. 📄

5.3. Combining xHTML & CSS with PHP & MySQL

The client-side of the site was developed before the server-side PHP and MySQL. This was done so that the final information that was required to be displayed on the site could be accounted for in the MySQL database and dynamic logic.

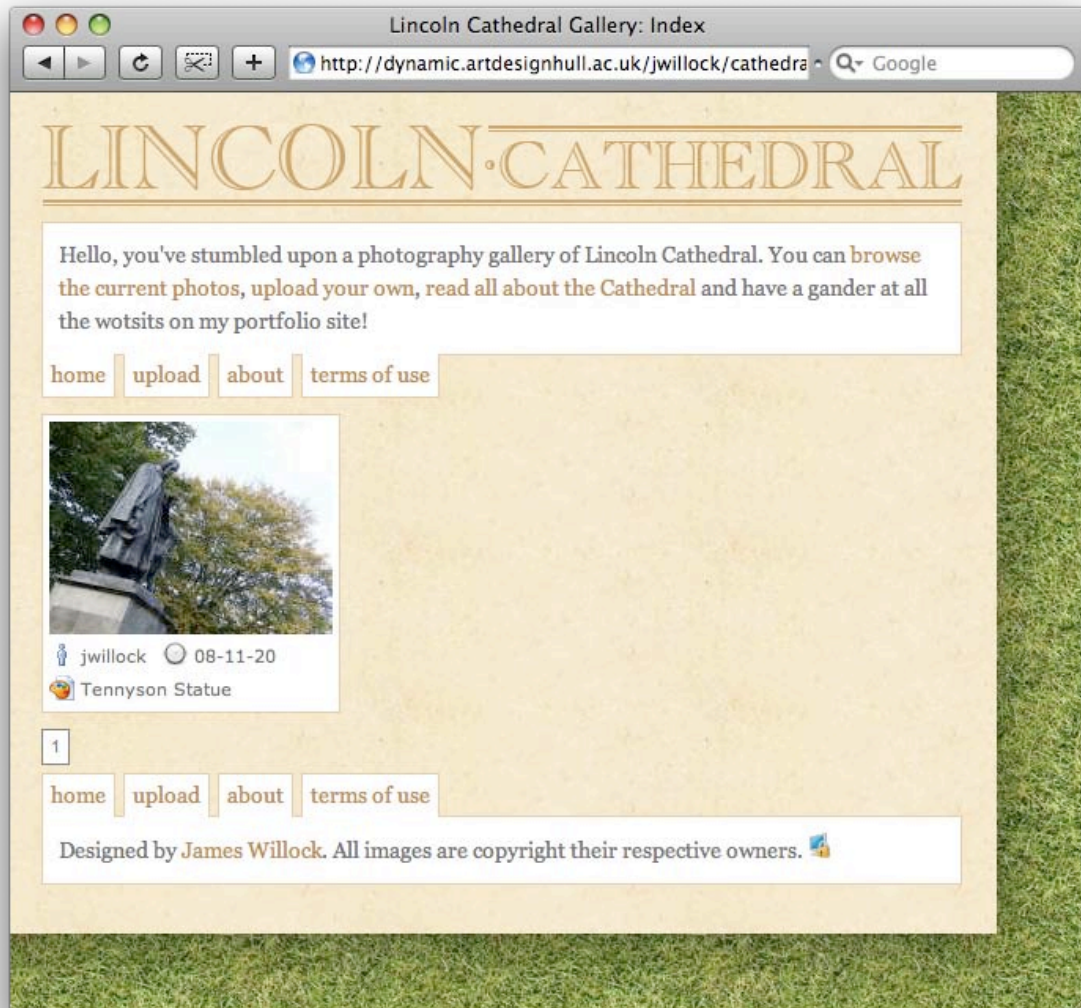
Using the classes and functions written for the site, "index.php" performs the following procedure:



5.4. Browser Testing

Before the site is deployed, it is important to ensure that the site renders properly using popular browsers and rendering engines. Using standards-based code should ensure that any rendering errors are kept to a minimum and that the site will display properly on a wide range of browsers.

5.4.1. Safari 3.2 (OS X 10.5.5)



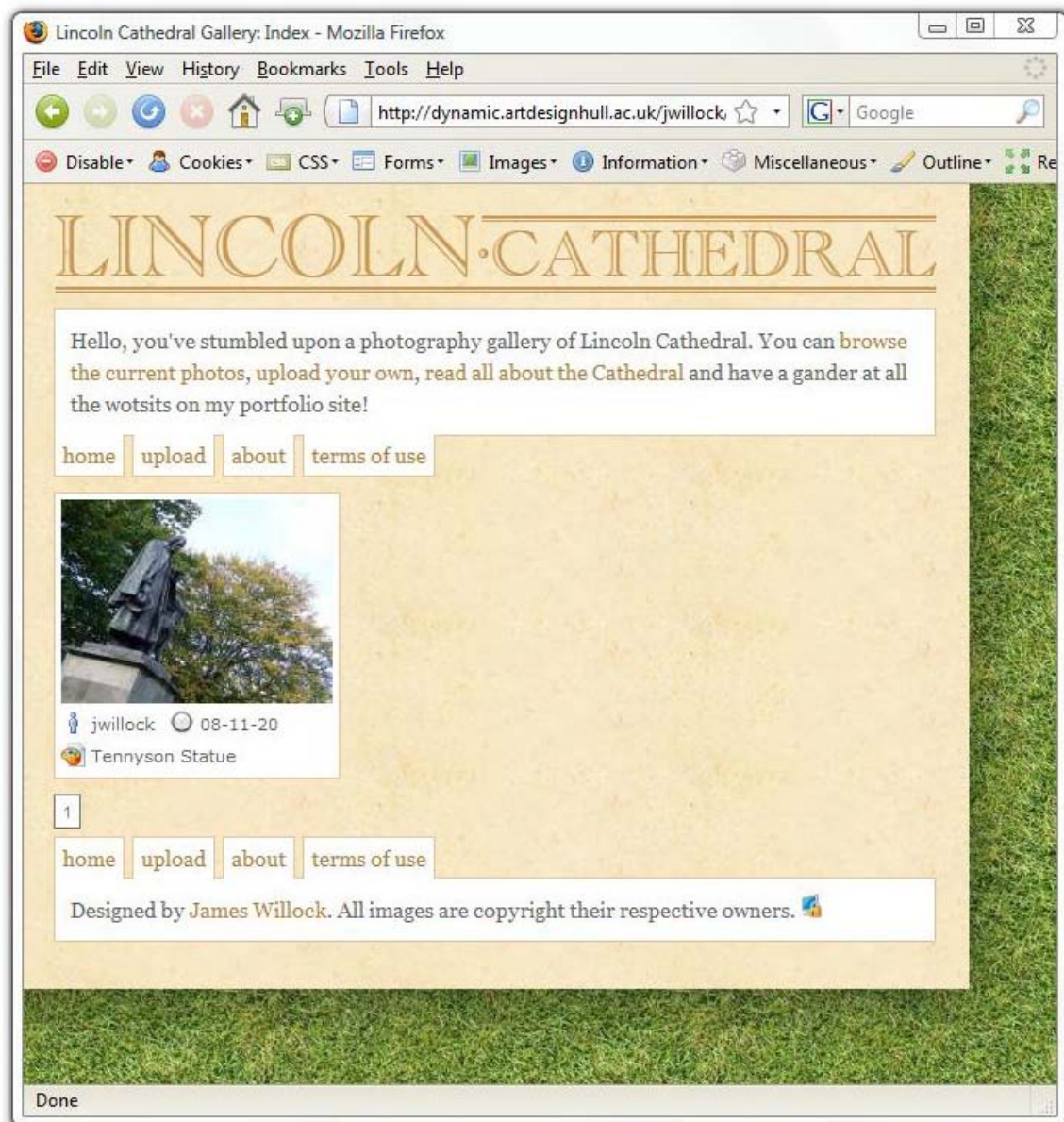
The site in Safari shows no obvious incorrect renderings on the home page, or other pages.

5.4.2. Camino 1.6.5 (OS X 10.5.5)



The site in Camino shows no obvious incorrect renderings on the home page, or other pages.

5.4.3. Firefox 3.0.4 (Windows Vista)



The site in Firefox shows no obvious incorrect renderings on the home page, or other pages.

5.5. Testing on Various Connection Speeds

Using the web site analyzer¹¹ at WebSiteOptimization, the speed at which a site should load on various connections can be determined. The Lincoln Cathedral Gallery home page returned the following results:

Connection Speed	Download Time
14.4k	194.58 seconds
28.8k	98.69 seconds
33.6k	84.99 seconds
56k	52.12 seconds
ISDN 128k	17.90 seconds
T1 1.44Mbps	4.11 seconds

The results displayed were not unsurprising. Since the site uses high-quality alpha-transparent PNG files for graphics, a Javascript library for client-side behaviour, and contains high-resolution user uploaded images. The download times cited are for first-time loading, and a user that views another page on the site will experience much faster load times due to browser cache of images and client-side scripts.

6. Conclusion

The produced web site fulfils the proposal outlined earlier in the document. The following objectives were set:

1. Home page listing several rows of the most recently uploaded images.
2. Pagination system for viewing pages of uploaded images.
3. "Detail" page for each image (dynamically generated), with fold out lightbox for high-resolution images.
4. Upload section allowing users to add their own images to the gallery.
5. A password-protected section allowing administrators to vet images, and accept or deny them.
6. Several information pages explaining the purpose and workings of the site.

The objectives set out in the project brief¹² have also been met, although there are definitely certain improvements or new features that would have been considered during a longer development cycle.

¹¹ WebSiteOptimization.com. <http://analyze.websiteoptimization.com/wso>

¹² NMH <http://www.newmediahull.lincoln.ac.uk/modules/IndBrief.php?briefId=8&year=2&courseId=3>

7. Recommendations

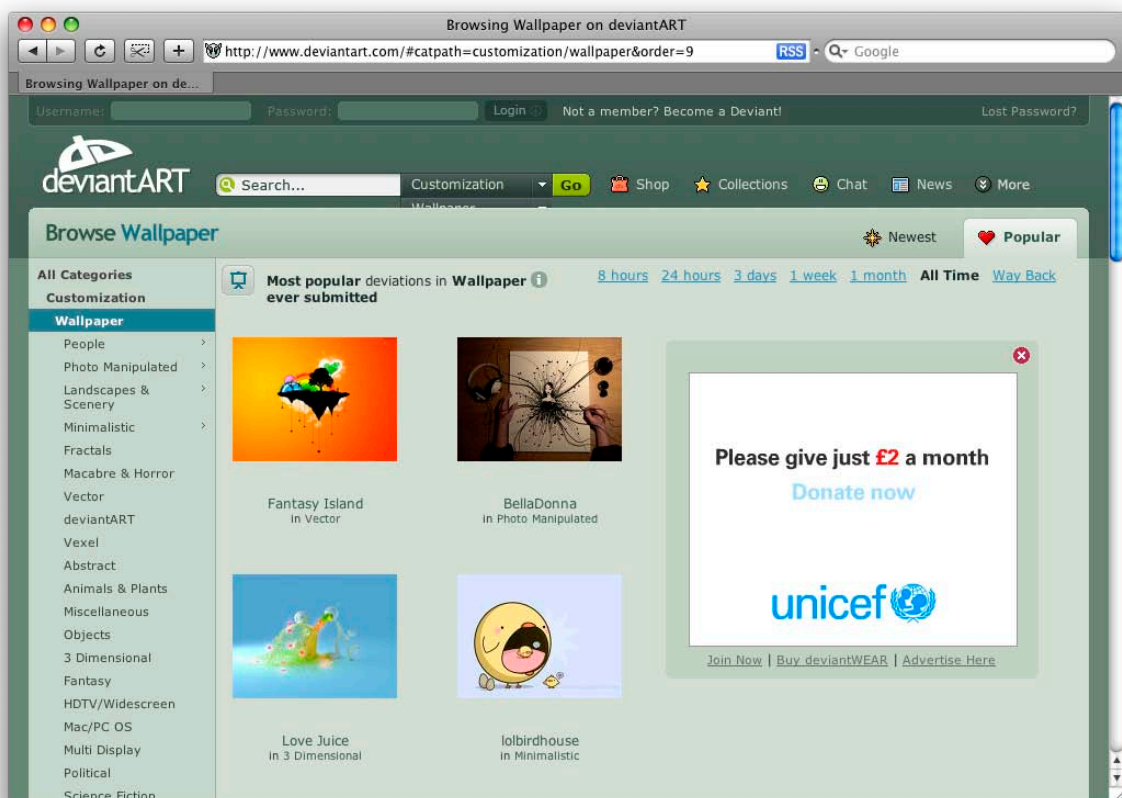
7.1. Improvements

If the development cycle had been longer, the following features would have complemented the site:

- Tagging system for attaching metadata to images on submission. This would have allowed the creation of a tag cloud to view images based on certain criteria. This would require the use of one or two new relational tables.
- More robust image uploading to allow the uploading of various formats of images.
- Commenting system to allow users to comment on specific images. This would require the use of a new relational table.
- Using “salt” in the hashing process to remove the danger of rainbow tables should the database be accessed by an unscrupulous user.

7.2. Proposal for a Different Dynamic Web Based Gallery

Had I decided not to develop a gallery for my photographs of Lincoln Cathedral, I would have considered developing a gallery for digitally developed computer wallpapers, a la DeviantArt¹³.



¹³ DeviantArt <http://www.deviantart.com>

7.3. Considering alternative programming language

If the site were to be developed on a server where PHP or MySQL were not present, we must consider the implications of developing the site using a different programming language or database server. Since the Windows operating system and its companion server technologies (Internet Information Services, or IIS, Microsoft SQL Server, and ASP.NET) are the major alternative on the hosted server market, they should be the primary alternative consideration.

The most obvious comparison is cost – Apache, PHP and MySQL are all “free” software (their unsupported releases do not require purchase or subscription costs), whereas Microsoft’s IIS and .NET parsers are bundled with its commercial business operating system Microsoft Windows Server. Microsoft SQL Server is software which ranges from free for a cut-down “express” version, to over £10 000 for its “Enterprise Edition”.

Additionally, we must consider performance. Since detailed benchmarks and objective analysis are difficult to find, the biggest performance factor is often the hardware of the server itself. On a budget, money spent on purchasing licenses for commercial software could not be put into dedicated server hardware, which may negatively impact performance.

One advantage of ASP.NET and its companion technologies is the Visual Studio IDE, which is a highly developed programming application for developing applications on Windows technologies. However, the Visual Studio application is another piece of commercial software which must be considered in the initial outlay.

8. Bibliography / References

Throughout the development of the gallery project, I used many different sources for research, code reference and testing. They are listed and categorized below.

8.1. Code references and libraries

- MooTools – a compact Javascript framework (<http://mootools.net/>)
- Smoothbox – a “Lightbox” script for MooTools (<http://gueschla.com/labs/smoothbox/>)
- PHP Documentation (<http://uk.php.net/manual/en/>)
- MySQL Documentation (<http://dev.mysql.com/doc/refman/4.1/en/>)

8.2. Website Testing

- W3C Markup Validation (<http://validator.w3.org/>)
- W3C CSS Validation (<http://jigsaw.w3.org/css-validator/>)
- The Web Site Optimizer (<http://www.websiteoptimization.com/services/analyze/>)

9. Appendices

9.1. Research

During the planning stages of the project, I used the web to research similar web sites to see how certain elements of the web site are used, what technologies are employed and how best to start the project. I used the following sites as case studies:

- Flickr¹⁴ - a massively popular photo-sharing website containing millions of images.
- DeviantArt¹⁵ - a large community gallery containing photographs and other digital images.
- Lincoln Cathedral Website¹⁶ – the official web site of the Cathedral, I used this site for information about the building to write content for the gallery.

¹⁴ Flickr <http://flickr.com/>

¹⁵ DeviantArt <http://www.deviantart.com>

¹⁶ Lincoln Cathedral <http://www.lincolncathedral.com/>

9.2. Code Listing

9.2.1. File: index.php

```
<?php
// James Willock: Lincoln Cathedral Gallery

// Note: Debug mode
error_reporting(E_ALL);

// Include functions and libraries
require('functions.php');

// Clean input incase Magic Quotes is off
$_POST = array_map('cleanData', $_POST);

// db connect
mysql_connect("localhost", "censored", "censored");
mysql_select_db("censored");

// start session
session_start();

// Begin templating
$thispage = new templatier();

if (isset($_GET['page'])) {
    $page = $_GET['page'];
} else {
    $page = "main";
}

switch ($page) {
    case "main":
        // Check for login data
        if (isset($_POST['luser_name']) && isset($_POST['lpass_word'])) {
            // hide password with secure hash
            if (galleryLogin($_POST['luser_name'], $_POST['lpass_word'])) {
                // logged in
                header("Location:
http://dynamic.artdesignhull.ac.uk/jwillock/cathedral/");
                die();
            }
        } else {
            // login successful, do stuff
            die("Incorrect details.");
        }
    }

    // Load content
    $gallery = new gallery();
    $thispage->set_file("front.tpl");
    if (isset($_GET['pagenum'])) {
        $thispage->add_tag("GALLERY", $gallery->
>displayImages($_GET['pagenum']));
    } else {
        $thispage->add_tag("GALLERY", $gallery->displayImages(1));
    }
    $thispage->finalise();
}
```

```

// Load header template, man!
$thispage->set_file("header.tpl");
$strap = new templatier();
$strap->set_file("strap.tpl");
$strap->add_tag("HIDDEN", "");
$strap->finalise();
$thispage->add_tag("STRAP", $strap->return_page());
if (isset($_SESSION['logged_in'])) {
    $loggedin = new templatier();
    $loggedin->set_file("nav-loggedin.tpl");
    $loggedin->add_tag("num", num_unapproved());
    $loggedin->add_tag("num_admins", num_unapprovedadmins());
    $loggedin->finalise();
    $thispage->add_tag("LOGGED_IN", $loggedin->return_page());
} else {
    $thispage->add_tag("LOGGED_IN", "");
}
$thispage->add_tag("TITLE", "Index");
// Set finalise() to 1 so header will be at top of page
$thispage->finalise(1);

// Load footer
$thispage->set_file("footer.tpl");
$thispage->add_tag("HIDDEN", "");
$thispage->finalise();

break; // end of main
case "unapproved":
// check for login
if (isset($_SESSION['logged_in'])) {
// Load content
    $gallery = new gallery();
    $thispage->set_file("front.tpl");
    if (isset($_GET['pagenum'])) {
    $thispage->add_tag("GALLERY", $gallery->
>displayImages($_GET['pagenum'], true));
    }
    else {
    $thispage->add_tag("GALLERY", $gallery->displayImages(1, true));
    }
    $thispage->finalise();

// Load header template, man!
$thispage->set_file("header.tpl");
$strap = new templatier();
$strap->set_file("strap-unapproved.tpl");
$strap->add_tag("HIDDEN", "");
$strap->finalise();
$thispage->add_tag("STRAP", $strap->return_page());
if (isset($_SESSION['logged_in'])) {
    $loggedin = new templatier();
    $loggedin->set_file("nav-loggedin.tpl");
    $loggedin->add_tag("num", num_unapproved());
    $loggedin->add_tag("num_admins", num_unapprovedadmins());
    $loggedin->finalise();
    $thispage->add_tag("LOGGED_IN", $loggedin->return_page());
} else {
    $thispage->add_tag("LOGGED_IN", "");
}
}

```

```

$thispage->add_tag("TITLE", "Unapproved");
// Set finalise() to 1 so header will be at top of page
$thispage->finalise(1);

// Load footer
$thispage->set_file("footer.tpl");
$thispage->add_tag("HIDDEN", "");
$thispage->finalise();
} else {
    header("Location:
http://dynamic.artdesignhull.ac.uk/jwillock/cathedral/");
    die();
}

break; // end of main

case "about":
// Load content
$thispage->set_file("about.tpl");
$thispage->add_tag("POPULAR", "");
$thispage->finalise();

// Load header template, man!
$thispage->set_file("header.tpl");
$strap = new templater();
$strap->set_file("about-strap.tpl");
$strap->add_tag("HIDDEN", "");
$strap->finalise();
$thispage->add_tag("STRAP", $strap->return_page());
if (isset($_SESSION['logged_in'])) {
    $loggedin = new templater();
    $loggedin->set_file("nav-loggedin.tpl");
    $loggedin->add_tag("num", num_unapproved());
    $loggedin->add_tag("num_admins", num_unapprovedadmins());
    $loggedin->finalise();
    $thispage->add_tag("LOGGED_IN", $loggedin->return_page());
} else {
    $thispage->add_tag("LOGGED_IN", "");
}
$thispage->add_tag("TITLE", "About");
// Set finalise() to 1 so header will be at top of page
$thispage->finalise(1);

// Load footer
$thispage->set_file("footer.tpl");
$thispage->add_tag("HIDDEN", "");
$thispage->finalise();

break; // end of main
case "browse":
// Load content
$thispage->set_file("browse.tpl");
$thispage->add_tag("USERS", listNames());
if (isset($_GET['names'])) {
    $thispage->add_tag("ENTRIES", entriesbyName($_GET['names']));
} else {
    $thispage->add_tag("ENTRIES", "");
}
if (isset($_GET['term'])) {
    $thispage->add_tag("SEARCH", searchEntries($_GET['term']));
    $thispage->add_tag("TERM", $_GET['term']);
}

```

```

    } else {
        $thispage->add_tag("SEARCH", "");
        $thispage->add_tag("TERM", "");
    }
    $thispage->finalise();

    // Load header template, man!
    $thispage->set_file("header.tpl");
    $strap = new templatier();
    $strap->set_file("strap.tpl");
    $strap->add_tag("HIDDEN", "");
    $strap->finalise();
    $thispage->add_tag("STRAP", $strap->return_page());
    if (isset($_SESSION['logged_in'])) {
        $loggedin = new templatier();
        $loggedin->set_file("nav-loggedin.tpl");
        $loggedin->add_tag("num", num_unapproved());
        $loggedin->add_tag("num_admins", num_unapprovedadmins());
        $loggedin->finalise();
        $thispage->add_tag("LOGGED_IN", $loggedin->return_page());
    } else {
        $thispage->add_tag("LOGGED_IN", "");
    }
    $thispage->add_tag("TITLE", "Search");
    // Set finalise() to 1 so header will be at top of page
    $thispage->finalise(1);

    // Load footer
    $thispage->set_file("footer.tpl");
    $thispage->add_tag("HIDDEN", "");
    $thispage->finalise();
break;
case "image":
    if (!isset($_GET['id'])) {
        die();
    }
    // Load content
    $gallery = new gallery();
    $thispage->set_file("front.tpl");
    $thispage->add_tag("GALLERY", $gallery->displaySingle($_GET['id']));
    $thispage->add_tag("POPULAR", "");
    $thispage->finalise();

    // Load header template, man!
    $thispage->set_file("header.tpl");
    $strap = new templatier();
    $strap->set_file("strap-image.tpl");
    $strap->add_tag("HIDDEN", "");
    $strap->finalise();
    $thispage->add_tag("STRAP", $strap->return_page());
    if (isset($_SESSION['logged_in'])) {
        $loggedin = new templatier();
        $loggedin->set_file("nav-loggedin.tpl");
        $loggedin->add_tag("num", num_unapproved());
        $loggedin->add_tag("num_admins", num_unapprovedadmins());
        $loggedin->finalise();
        $thispage->add_tag("LOGGED_IN", $loggedin->return_page());
    } else {
        $thispage->add_tag("LOGGED_IN", "");
    }
    $thispage->add_tag("TITLE", "Image");

```

```

// Set finalise() to 1 so header will be at top of page
$thispage->finalise(1);

// Load footer
$thispage->set_file("footer.tpl");
$thispage->add_tag("HIDDEN", "");
$thispage->finalise();

break; // end of main
case "upload":
    $thispage->set_file("upload.tpl");
    $thispage->add_tag("POPULAR", "");

    if (!empty($_FILES['uploadfile'])) {
        if (empty($_POST['image_name']) || empty($_POST['user_name']))
        {
            // input
            $error = new templater();
            $error->set_file("error.tpl");
            $error->add_tag("HIDDEN", "");
            $error->finalise();
            $thispage->add_tag("ERROR", $error->return_page());
        }
        else {
            // Files detected, try upload
            $uploadimage = new imagestuff();
            $uploadimage->setTimestamp();
            $uploadimage->formatImage("default");
            $uploadimage->formatImage("thumbnail");
            $uploadimage->formatImage("highres");
            $uploadimage->createEntry($_POST['image_name'], $uploadimage-
>checkUsername($_POST['user_name'], $_POST['description']));
            // Successful, display message
            $error = new templater();
            $error->set_file("successful.tpl");
            $error->add_tag("HIDDEN", "");
            $error->finalise();
            $thispage->add_tag("ERROR", $error->return_page());
        }
    }
    else {
        $thispage->add_tag("ERROR", "");
    }

    $thispage->finalise();

// Load header template, man!
$thispage->set_file("header.tpl");
$strap = new templater();
$strap->set_file("strap-upload.tpl");
$strap->add_tag("HIDDEN", "");
$strap->finalise();
$thispage->add_tag("STRAP", $strap->return_page());
if (isset($_SESSION['logged_in'])) {
    $loggedin = new templater();
    $loggedin->set_file("nav-loggedin.tpl");
    $loggedin->add_tag("num", num_unapproved());
    $loggedin->add_tag("num_admins", num_unapprovedadmins());
    $loggedin->finalise();
    $thispage->add_tag("LOGGED_IN", $loggedin->return_page());
}

```

```

    } else {
        $thispage->add_tag("LOGGED_IN", "");
    }
    $thispage->add_tag("TITLE", "Upload");
    // Set finalise() to 1 so header will be at top of page
    $thispage->finalise(1);

    // Load footer
    $thispage->set_file("footer.tpl");
    $thispage->add_tag("HIDDEN", "");
    $thispage->finalise();

break; // end of main
case "login":
    // Load content
    if (!isset($_SESSION['logged_in'])) {
        $thispage->set_file("login.tpl");
        $thispage->add_tag("POPULAR", "");
        $thispage->finalise();
    } else {
        $thispage->set_file("already-loggedin.tpl");
        $thispage->add_tag("POPULAR", "");
        $thispage->finalise();
    }
    // Load header template, man!
    $thispage->set_file("header.tpl");
    $strap = new templater();
    $strap->set_file("strap.tpl");
    $strap->add_tag("HIDDEN", "");
    $strap->finalise();
    $thispage->add_tag("STRAP", $strap->return_page());
    if (isset($_SESSION['logged_in'])) {
        $loggedin = new templater();
        $loggedin->set_file("nav-loggedin.tpl");
        $loggedin->add_tag("num", num_unapproved());
        $loggedin->add_tag("num_admins", num_unapprovedadmins());
        $loggedin->finalise();
        $thispage->add_tag("LOGGED_IN", $loggedin->return_page());
    } else {
        $thispage->add_tag("LOGGED_IN", "");
    }
    $thispage->add_tag("TITLE", "Login");
    // Set finalise() to 1 so header will be at top of page
    $thispage->finalise(1);

    // Load footer
    $thispage->set_file("footer.tpl");
    $thispage->add_tag("HIDDEN", "");
    $thispage->finalise();

break; // end of main
case "adminform":
    // Load content
    $thispage->set_file("adminform.tpl");
    if (isset($_POST['user_name']) && isset($_POST['pass_word'])) {
        // adminform filled out, process
        if (!isset($_POST['email_address'])) {$_POST['email_address'] = "";}
        addAdministrator($_POST['user_name'], $_POST['email_address'],
$_POST['pass_word']);
        $thispage->add_tag("COMPLETED", "Your entry has been received!");
    } else {

```

```

        $thispage->add_tag("COMPLETED", "");
    }
    $thispage->finalise();

    // Load header template, man!
    $thispage->set_file("header.tpl");
    $strap = new templater();
    $strap->set_file("strap.tpl");
    $strap->add_tag("HIDDEN", "");
    $strap->finalise();
    $thispage->add_tag("STRAP", $strap->return_page());
    if (isset($_SESSION['logged_in'])) {
        $loggedin = new templater();
        $loggedin->set_file("nav-loggedin.tpl");
        $loggedin->add_tag("num", num_unapproved());
        $loggedin->add_tag("num_admins", num_unapprovedadmins());
        $loggedin->finalise();
        $thispage->add_tag("LOGGED_IN", $loggedin->return_page());
    } else {
        $thispage->add_tag("LOGGED_IN", "");
    }
    $thispage->add_tag("TITLE", "Admin registration form");
    // Set finalise() to 1 so header will be at top of page
    $thispage->finalise(1);

    // Load footer
    $thispage->set_file("footer.tpl");
    $thispage->add_tag("HIDDEN", "");
    $thispage->finalise();
    break; // end of admin form
    case "unapprovedadmins":
        if (!isset($_SESSION['logged_in'])) {
            header("Location:
http://dynamic.artdesignhull.ac.uk/jwillock/cathedral/");
            die();
        }
        if (isset($_GET['id'])) {
            approveAdmin($_GET['id']);
        }
        // Load content
        $thispage->set_file("unapprovedadmins.tpl");
        $thispage->add_tag("UNAPPROVED", unapprovedAdmins());
        $thispage->finalise();

        // Load header template, man!
        $thispage->set_file("header.tpl");
        $strap = new templater();
        $strap->set_file("strap.tpl");
        $strap->add_tag("HIDDEN", "");
        $strap->finalise();
        $thispage->add_tag("STRAP", $strap->return_page());
        if (isset($_SESSION['logged_in'])) {
            $loggedin = new templater();
            $loggedin->set_file("nav-loggedin.tpl");
            $loggedin->add_tag("num", num_unapproved());
            $loggedin->add_tag("num_admins", num_unapprovedadmins());
            $loggedin->finalise();
            $thispage->add_tag("LOGGED_IN", $loggedin->return_page());
        } else {
            $thispage->add_tag("LOGGED_IN", "");
        }
    }
}

```

```

$thispage->add_tag("TITLE","Unapproved admins");
// Set finalise() to 1 so header will be at top of page
$thispage->finalise(1);

// Load footer
$thispage->set_file("footer.tpl");
$thispage->add_tag("HIDDEN","");
$thispage->finalise();
break; // end of admin form
case "logout":
    session_destroy();
    header("Location:
http://dynamic.artdesignhull.ac.uk/jwillock/cathedral/");
    die();
break; // end of logout
case "deleteimage":
    if (isset($_GET['imageid']) && isset($_SESSION['logged_in'])) {
        deleteimage($_GET['imageid']);
    }
    else {
        header("Location:
http://dynamic.artdesignhull.ac.uk/jwillock/cathedral/");
        die();
    }
break; // end of delete image
case "approve":
    if (isset($_GET['imageid']) && isset($_SESSION['logged_in'])) {
        approveimage($_GET['imageid']);
    }
    else {
        header("Location:
http://dynamic.artdesignhull.ac.uk/jwillock/cathedral/");
        die();
    }
break; // end of delete image
default:
    die();
}

// Print page, man.
echo $thispage->return_page();

?>

```

9.2.2. File: functions.php

```
<?php

// Templating class and functions
class templater {
    var $clay_string;
    var $final_array = array();

    function set_file($file) {
        $this->clay_string = file_get_contents("./templates/" . $file);
    }

    function add_tag($tag,$replace) {
        $this->clay_string = str_replace("_" . $tag . "_", $replace,
$this->clay_string);
    }

    function finalise($position = 0) {
        if ($position == 1) {
            // place at beginning of array
            array_unshift($this->final_array, $this->clay_string);
        }
        else {
            $this->final_array[] = $this->clay_string;
        }
    }

    function return_page() {
        $final = "";
        foreach ($this->final_array as $val) {
            $final .= $val;
        }
        return $final;
    }
}

// End of templating class

function cleanData($data) {
    return htmlentities($data, ENT_QUOTES, 'UTF-8');
}

class imagestuff {

    var $imageurl;
    var $timestamp;

    function setTimestamp() {
        $this->timestamp = time();
        $this->imageurl = $this->timestamp . $_FILES['uploadfile']['name'];
    }

    function formatImage($width) {
        switch ($width) {
            case "highres":
                $uploadedimage = $_FILES['uploadfile']['tmp_name'];

```

```

        $newimage = imagecreatefromjpeg($uploadedimage);
        list($imagewidth,$imageheight) =
getimagesize($uploadedimage);
        $siteimagewidth = 1280;
        $siteimageheight = ($imageheight / $imagewidth) *
$siteimagewidth;
        $temporaryimage =
imagecreatetruecolor($siteimagewidth,$siteimageheight);

        imagecopyresampled($temporaryimage,$newimage,0,0,0,0,$siteimagewidth,$
siteimageheight,$imagewidth,$imageheight);
        $filename = "./userimages/highres/" . $this->timestamp .
$_FILES['uploadfile']['name'];
        imagejpeg($temporaryimage,$filename,60);
        imagedestroy($newimage);
        imagedestroy($temporaryimage);
        return true;
    break;
    case "thumbnail":
        $uploadedimage = $_FILES['uploadfile']['tmp_name'];
        $newimage = imagecreatefromjpeg($uploadedimage);
        list($imagewidth,$imageheight) =
getimagesize($uploadedimage);
        $siteimagewidth = 180;
        $siteimageheight = ($imageheight / $imagewidth) *
$siteimagewidth;
        $temporaryimage =
imagecreatetruecolor($siteimagewidth,$siteimageheight);

        imagecopyresampled($temporaryimage,$newimage,0,0,0,0,$siteimagewidth,$
siteimageheight,$imagewidth,$imageheight);
        $filename = "./userimages/thumbnails/" . $this->timestamp
. $_FILES['uploadfile']['name'];
        imagejpeg($temporaryimage,$filename,60);
        imagedestroy($newimage);
        imagedestroy($temporaryimage);
        return true;
    break;
    case "default":
        $uploadedimage = $_FILES['uploadfile']['tmp_name'];
        $newimage = imagecreatefromjpeg($uploadedimage);
        list($imagewidth,$imageheight) =
getimagesize($uploadedimage);
        $siteimagewidth = 512;
        $siteimageheight = ($imageheight / $imagewidth) *
$siteimagewidth;
        $temporaryimage =
imagecreatetruecolor($siteimagewidth,$siteimageheight);

        imagecopyresampled($temporaryimage,$newimage,0,0,0,0,$siteimagewidth,$
siteimageheight,$imagewidth,$imageheight);
        $filename = "./userimages/" . $this->timestamp .
$_FILES['uploadfile']['name'];
        imagejpeg($temporaryimage,$filename,60);
        imagedestroy($newimage);
        imagedestroy($temporaryimage);
        return true;
    break;
}

```

```

    }

    function checkUsername($username) {
        // first, check if the username is already in use. if so, return the
        id of the name. if not, add the name to the table, and return the id.
        $result = mysql_query("SELECT * FROM lc_names WHERE names_name = '" .
$username . "' LIMIT 1");
        if (mysql_num_rows($result) == 1) {
            // record found, fetch id
            while ($row = mysql_fetch_object($result)) {
                $nameid = $row->names_id;
            }
        } else {
            // no record found, insert name and get id
            mysql_query("INSERT INTO lc_names (names_name) VALUES ('" .
$username . "')");
            $nameid = mysql_insert_id();
        }
        return $nameid;
    }

    function createEntry ($image_name, $nameid, $description) {
        $query = "INSERT INTO lc_photos (image_name, name, description,
image_url) VALUES ('" . $image_name . "', '" . $nameid . "', '" .
$description . "', '" . $this->imageurl . "')";
        mysql_query($query);
        echo mysql_error();
    }
}

class gallery {
// class for fetching data from db

    function displayImages($pagenum, $unapproved = false) {
        $gallery = new templater();

        // Pagination! Fetch number of rows, then do maths.
        if ($unapproved) {
            $rows = mysql_num_rows(mysql_query("SELECT * FROM lc_photos
LEFT JOIN lc_names ON lc_photos.name = lc_names.names_id WHERE processed =
0"));
            if ($rows == 0) {return "No images!";}
        }
        else {
            $rows = mysql_num_rows(mysql_query("SELECT * FROM lc_photos
LEFT JOIN lc_names ON lc_photos.name = lc_names.names_id WHERE processed =
1"));
            if ($rows == 0) {return "No images!";}
        }
        // This tells us the page number of our last page
        $page_rows = 6;
        $last = ceil($rows / $page_rows);

        //this makes sure the page number isn't below one, or more than our
maximum pages
        if ($pagenum < 1) {
            $pagenum = 1;
        }

        elseif ($pagenum > $last) {

```

```

        $pagenum = $last;
    }

    //This sets the range to display in our query
    $max = 'limit ' . ($pagenum - 1) * $page_rows . ',' . $page_rows;
    if ($unapproved == true) {
        $result = mysql_query("SELECT * FROM lc_photos LEFT JOIN
lc_names ON lc_photos.name = lc_names.names_id WHERE processed = 0 ORDER BY
id DESC ". $max);
    }
    else {
        $result = mysql_query("SELECT * FROM lc_photos LEFT JOIN
lc_names ON lc_photos.name = lc_names.names_id WHERE processed = 1 ORDER BY
id DESC ". $max);
    }
    echo mysql_error();

    while ($row = mysql_fetch_object($result)) {
        $gallery->set_file("gallery.tpl");
        $gallery->add_tag("id", $row->id);
        $gallery->add_tag("image_name", $row->image_name);
        $gallery->add_tag("user_name", $row->names_name);
        $gallery->add_tag("name_id", $row->name);
        $gallery->add_tag("description", $row->description);
        $gallery->add_tag("image_url", $row->image_url);
        $gallery->add_tag("timestamp", substr(date("d/m/y, $row-
>timestamp"), 0, 8));

        if ($unapproved == true) {
            $approve = new templater();
            $approve->set_file("approvedeny.tpl");
            $approve->add_tag("ID", $row->id);
            $approve->finalise();
            $gallery->add_tag("approvedeny", $approve-
>return_page());
        } else {
            $gallery->add_tag("approvedeny", "");
        }

        $gallery->finalise();
    }

    $gallery->set_file("pagenav-holder.tpl");
    if ($unapproved) {
        $paged = new templater();
        for ($i=1; $i <= $last; $i++) {
            if($i == $pagenum) {
                $paged->set_file("pagenav-unapproved.tpl");
                $paged->add_tag("url", $i);
                $paged->add_tag("current", "current");
                $paged->finalise();
            }

            else {
                $paged->set_file("pagenav-unapproved.tpl");
                $paged->add_tag("url", $i);
                $paged->add_tag("current", "");
                $paged->finalise();
            }
        }
    }
}

```

```

    } else {
        $paged = new templatier();
        for ($i=1; $i <= $last; $i++) {
            if($i == $pagenum) {
                $paged->set_file("pagenav.tpl");
                $paged->add_tag("url", $i);
                $paged->add_tag("current", "current");
                $paged->finalise();
            }

            else {
                $paged->set_file("pagenav.tpl");
                $paged->add_tag("url", $i);
                $paged->add_tag("current", "");
                $paged->finalise();
            }
        }
    }
    $gallery->add_tag("PAGES", $paged->return_page());
    $gallery->finalise();
    return $gallery->return_page();
}

function displaySingle($id) {
    $gallery = new templatier();
    $result = mysql_query("SELECT * FROM lc_photos LEFT JOIN lc_names ON
lc_photos.name = lc_names.names_id WHERE id = '" . $id . "' LIMIT 1");
    while ($row = mysql_fetch_object($result)) {
        $gallery->set_file("image.tpl");
        $gallery->add_tag("id", $row->id);
        $gallery->add_tag("image_name", $row->image_name);
        $gallery->add_tag("user_name", $row->names_name);
        $gallery->add_tag("description", $row->description);
        $gallery->add_tag("image_url", $row->image_url);
        $gallery->add_tag("next_previous", $this-
>displayNextPrevious($row->id));
        $gallery->add_tag("timestamp", substr(date("d/m/y, $row-
>timestamp"), 0, 8));
        if (isset($_SESSION['logged_in'])) {
            $approve = new templatier();
            $approve->set_file("delete.tpl");
            $approve->add_tag("id", $row->id);
            $approve->finalise();
            $gallery->add_tag("delete", $approve->return_page());
        } else {
            $gallery->add_tag("delete", "");
        }
        $gallery->finalise();
    }
    return $gallery->return_page();
}

function displayNextPrevious($id) {
    $displaynextprevious = new templatier();
    $result = mysql_query("SELECT * FROM lc_photos LEFT JOIN lc_names ON
lc_photos.name = lc_names.names_id WHERE processed = 1 AND id < '" . $id .
"' ORDER BY id DESC LIMIT 1");
    while ($row = mysql_fetch_object($result)) {
        $displaynextprevious->set_file("nav-previous.tpl");
    }
}

```

```

        $displaynextprevious->add_tag("id", $row->id);
        $displaynextprevious->add_tag("image_name", $row->image_name);
        $displaynextprevious->add_tag("user_name", $row->names_name);
        $displaynextprevious->add_tag("description", $row-
>description);
        $displaynextprevious->add_tag("image_url", $row->image_url);
        $displaynextprevious->finalise();
    }
    $result = mysql_query("SELECT * FROM lc_photos LEFT JOIN lc_names ON
lc_photos.name = lc_names.names_id WHERE processed = 1 AND id > '" . $id .
"' ORDER BY id ASC LIMIT 1");
    while ($row = mysql_fetch_object($result)) {
        $displaynextprevious->set_file("nav-next.tpl");
        $displaynextprevious->add_tag("id", $row->id);
        $displaynextprevious->add_tag("image_name", $row->image_name);
        $displaynextprevious->add_tag("user_name", $row->names_name);
        $displaynextprevious->add_tag("description", $row-
>description);
        $displaynextprevious->add_tag("image_url", $row->image_url);
        $displaynextprevious->finalise();
    }
    return $displaynextprevious->return_page();
}

}

function deleteimage($id) {
    mysql_query("DELETE FROM lc_photos WHERE id = '" . $id . "' LIMIT 1");
    header("Location:
http://dynamic.artdesignhull.ac.uk/jwillock/cathedral/");
    die();
}

function approveimage($id) {
    mysql_query("UPDATE lc_photos SET processed = '1' WHERE id = '" . $id
. "' LIMIT 1");
    header("Location:
http://dynamic.artdesignhull.ac.uk/jwillock/cathedral/");
    die();
}

function num_unapproved() {
    $result = mysql_query("SELECT * FROM lc_photos WHERE processed = 0");
    return mysql_num_rows($result);
}

function num_unapprovedadmins() {
    $result = mysql_query("SELECT * FROM lc_admins WHERE active = 0");
    return mysql_num_rows($result);
}

function getName($id) {
    // returns the name of a user, requires user id
    $result = mysql_query("SELECT * FROM lc_names WHERE names_id = '" .
$id . "' LIMIT 1");
    while ($row = mysql_fetch_object($result)) {
        return $row->names_name;
    }
}

```

```

function entriesbyName($id) {
    // lists all the entries by a user based on id
    $entryholder = new templater();
    $entryholder->set_file("entryholder.tpl");
    $entries = new templater();
    $result = mysql_query("SELECT * FROM lc_photos LEFT JOIN lc_names ON
lc_photos.name = lc_names.names_id WHERE processed = 1 AND name = '" . $id
. "' ORDER BY timestamp DESC");
    while ($row = mysql_fetch_object($result)) {
        $entries->set_file("entries.tpl");
        $entries->add_tag("id", $row->id);
        $entries->add_tag("image_name", $row->image_name);
        $entries->add_tag("user_name", $row->names_name);
        $entries->add_tag("description", $row->description);
        $entries->add_tag("image_url", $row->image_url);
        $entries->finalise();
    }
    $entryholder->add_tag("NAME", getName($id));
    $entryholder->add_tag("ENTRIES", $entries->return_page());
    $entryholder->finalise();
    return $entryholder->return_page();
}

function listNames() {
    // lists all the users in the db
    $entries = new templater();
    $result = mysql_query("SELECT * FROM lc_names ORDER BY names_id
ASC");
    while ($row = mysql_fetch_object($result)) {
        $entries->set_file("names.tpl");
        $entries->add_tag("names_id", $row->names_id);
        $entries->add_tag("names_name", $row->names_name);
        $entries->finalise();
    }
    return $entries->return_page();
}

function searchEntries($term) {
    // lists all the users in the db
    $search = new templater();
    $result = mysql_query("SELECT * FROM lc_photos LEFT JOIN lc_names ON
lc_photos.name = lc_names.names_id WHERE image_name LIKE '%$term%'");
    if (mysql_num_rows($result) == 0) {
        return "<p>You searched for <strong>" . $term . "</strong>.
Nothing found, sorry!</p>";
    }
    while ($row = mysql_fetch_object($result)) {
        $search->set_file("searchresults.tpl");
        $search->add_tag("id", $row->id);
        $search->add_tag("image_name", $row->image_name);
        $search->add_tag("description", $row->description);
        $search->add_tag("user_name", $row->names_name);
        $search->add_tag("image_url", $row->image_url);
        $search->finalise();
    }
    return $search->return_page();
}

function addAdministrator ($user_name, $email_address, $pass_word) {
    $query = "INSERT INTO lc_admins (admins_name, admins_email,

```

```

admins_password) VALUES ('" . $user_name . "', '" . $email_address . "',
'" . sha1($pass_word) . "');"
    mysql_query($query);
}

function galleryLogin($username, $password) {
    $result = mysql_query("SELECT * FROM lc_admins WHERE admins_name = '"
. $username . "' AND admins_password = '" . sha1($password) . "' AND active
= 1 LIMIT 1");
    if (mysql_num_rows($result) == 1) {
        // details accepted, set login.
        $_SESSION['logged_in'] = true;
        return true;
    } else {
        // details incorrect, throw error.
        return false;
    }
}

function unapprovedAdmins() {
    // lists all the users in the db
    $entries = new templatere();
    $result = mysql_query("SELECT * FROM lc_admins WHERE active = 0 ORDER
BY admins_id ASC");
    if (mysql_num_rows($result) == 0) {
        return "<p>No unapproved admins.</p>";
    }
    while ($row = mysql_fetch_object($result)) {
        $entries->set_file("admins.tpl");
        $entries->add_tag("admins_id", $row->admins_id);
        $entries->add_tag("admins_name", $row->admins_name);
        $entries->add_tag("admins_email", $row->admins_email);
        $entries->finalise();
    }
    return $entries->return_page();
}

function approveAdmin($id) {
    $query = "UPDATE lc_admins SET active = '1' WHERE admins_id = " . $id
. " LIMIT 1";
    mysql_query($query);
}

?>

```