WIGI
(Webbased Image Gallery Interface)
Department of Computing Science
8th June 2005

Martin Rölin, Axel Seugling, Johan Noren
{c01mrn, c01asg, c01jn}@cs.umu.se

Find WIGI here:
http://www.cs.umu.se/~c01mrn/db2/project

Abstract

This report describes design and functionality of a web based image gallery WIGI. Some features of the gallery is to upload images, search images, sort images and download galleries. The system uses PHP, JavaScripts and PostgreSQL to implement the above features. WIGI is a full featured image gallery and will in the future be used by people who like to share there images.

1 Introduction

WIGI\(^1\) is a web portal for storing and sharing images. WIGI allows administrators and registered users to maintain an image gallery supporting hierarchical organization of galleries and images. WIGI allows registered users to:

- Upload images. Supports packed archives of images in different formats.
- Search uploaded images and galleries. Searches can be performed on many different attributes or combinations of these.
- Sort images in galleries. Images can be positioned in different orders within galleries. Images can also be moved between galleries.
- Download galleries. There are two ways to download galleries, either as compressed archives or as transmission to an external FTP server.
- A user can mark their images as non downloadable then other users can’t download them.
- Grade and commenting images. When browsing the galleries, users are able to grade or comment images while watching other users comments and grades as well.

\(^1\)Webbased Image Gallery Interface
• See different statistics.

Users that aren’t registered can view, grade, comment and search for images.

2 Approach

To make WIGI easy accessible, the gallery is web based using XHTML combined with CSS. The pages are generated by PHP running on top a PostgreSQL database. The database related code is modularized and should be runnable on all SQL compliant database systems.

2.1 PHP/HTML layer

When scripting pages, a functional approach has been deployed, though the possibility to use object orientation in PHP has been used to some extent. In this way classes has been created for often used entities such as database connections and up/download mechanisms.

The SESSION feature of PHP has been used to create lasting sessions for users. This technique makes it possible to login and authenticate towards the server while browsing the site. It is also practical to store data such as search results within the session to save recomputations.

The layout of the site is based on stylesheets. All graphical attributes can be adjusted in style.css.

2.2 Database layer

The database have been developed in PostgreSQL, because this offers a stable database with all the features needed in the project. The database is rather simple, mainly built around the relation Image. The attribute pgallery in the relation Gallery recursively points back to the relation, saying that a gallery is a subgallery to another Gallery. The attribute image in Gallery states the image to represent the gallery. The following database schema shows in an UML like notion how the database have been built. The arrows in figure 1 shows foreign keys in the database. The datatype serial is often used in the database to represent index. This is a easy way of getting a unique attribute for each tuple.

3 Result

The final result met the requirements that were decided upon during the planning phase of the project. To these a set of extra features has been added during the projects development. A lot of time has been spent on the graphical user interface which in our opinion is both aesthetic and intuitive. When evaluating
the result of the project the appearance of the web site is as important as the functionality if it should have some practical use.

From a performance perspective the system has not shown any bottlenecks yet. This is however hard to predict before larger amounts of images and data has been added to the database. Since images are not stored within the database large amounts of images should scale well. What could become slow is the advanced search option provided where users can make searches on many different fields with substring matching. The LIKE feature of PostgreSQL has been used to make these case insensitive matches which performance is unknown.
3.1 Screenshots

The following screenshots show some of the gallery functionality.

![Screenshot of gallery view](image)

Figure 2: A gallery view. Images are listed after one another. One page in the gallery fits 16 images, if there are more than 16 images the gallery will be divided in several pages.

<table>
<thead>
<tr>
<th>Username:</th>
<th>admin</th>
<th>Upload: 32.1%</th>
<th>Views: 56.7%</th>
<th>Avg. rating: 4.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>admin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uploads:</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Views:</td>
<td>195</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recv. coments:</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Made coments:</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Allow downloads:</td>
<td>False</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Username:</th>
<th>Martin</th>
<th>Upload: 55.6%</th>
<th>Views: 38.7%</th>
<th>Avg. rating: 3.7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>Martin Rölin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uploads:</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Views:</td>
<td>133</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recv. coments:</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Made coments:</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Allow downloads:</td>
<td>False</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3: Some statistics, e.g. showing that admin have uploaded 32.1% of the images uploaded, 56.7% of the images that have been viewed are uploaded by admin, average rating of admins images are 4.3.

4 Discussion

Making a good and well written image gallery is not a thing you produce in a couple of weeks, to make this gallery to be competitive with existing galleries
a little more work has to be done.

4.1 Problems

One big problem was to make the drag and drop function work well in both Mozilla Firefox and Internet Explorer due to difference in handling of stylesheets and javascript. Besides from that the project has not given us any particularly problematic tasks.

4.2 Future work

WIGI will be used in the future by members in the lab group. There are some issues that has not yet been resolved or implemented. For example some mechanism to remove galleries by administrators and possibilities to assign an image to several galleries. For larger communities it should also be nice with some sort of grouping of users to limit access rights to images within the group. Another limitation is the upload mechanism that currently only supports archives of images. There should be some way do upload single images as well.

An important issue to deal with is the security flaws in the current design. All strings posted from pages should be escaped or made as prepared statements so that evil users cannot create malicious SQL queries in form fields.

There is also an issue with losing sessions when transmitting data over xmlhttp. Xmlhttp opens a separate http connection which the web server cannot securely authenticate with the already opened session.

5 Conclusions

The project has shown how to create an advanced web application combining databases and serverside pages. Some effort has also been put into making a nice graphical users interface that is intuitive for the user.

We are pleased with the result of this project. WIGI fulfills most demands you can have on an image gallery and will be used by us in the future.

6 Credits

The JavaScript for sorting images by dragging them in place have been developed by Toolman and adjusted by Samuel Sjöberg. Also thanks to Samuel Bengtsson for help with interface and design issues.