|  |
| --- |
| University of windsor |
| UWindsor Foreigners Blog Documentation |
| **60-334 World Wide Web Information Systems Development Professor Paul Preney** |
|  |
| Felipe Ramon de Lara, Flavien Bossiaux, Paulo Cardoso and Xuan Yao |
| **04/10/2014** |

|  |
| --- |
|  |

Index

[How install 2](#_Toc385100557)

[Website Structure 2](#_Toc385100558)

[Architecture 3](#_Toc385100559)

[Modules 4](#_Toc385100560)

[Login 5](#_Toc385100561)

[Register 5](#_Toc385100562)

[New Post 6](#_Toc385100563)

[Edit Post 7](#_Toc385100564)

[Delete Post 7](#_Toc385100565)

[About us 7](#_Toc385100566)

[Contact Us 8](#_Toc385100567)

**Introduction**

The UWindsor Foreigners Blog was conceived for international students exchange experiences about their lives at the University of Windsor. With its simple and clean layout that provides an intuitive usability for users. After you create your User is enabled to interact with the other exchange students can leave messages, edit your messages and deletes them.

# How install

Our project has an installer that prepares all requirements which enable the availability of UWindsor Foreigners Blog to the use. This installer is the file install.php that is in the folder public\_html. To install de blog copy all the files to the folder public html on your server. Then type on the URL bar on your browser “http://www.yourSite.com/install.php”.

The file install.php is responsible to create the database with his tables and configurations to the use. MyAdmin installed on the server is needed. After that the blog is installed and ready for use.

# Website Structure

The website is set up in a hierarchical format.

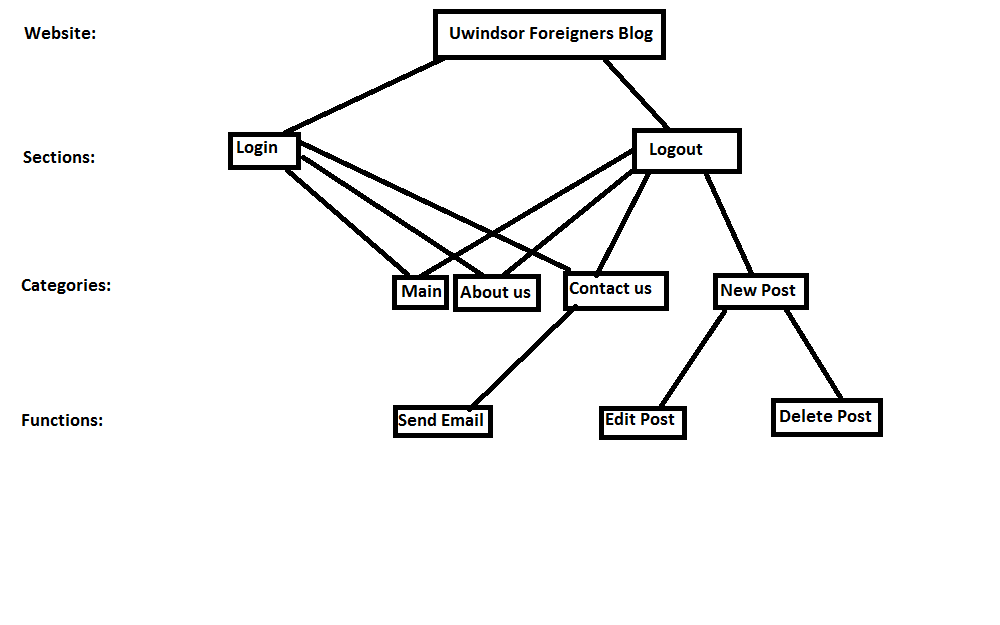


Figure 1 Website Structure Diagram

# Architecture

The project can be divided on three layers on the model MVC. And the project provides support for using or not of javascript. All the managers classes and static objects (i.e. config.php file of database settings), the folder classes/ is where these files are located. The folder includes/ is responsible for hold files that are sub part of a page that are not supposed to be accessed by URL (i.e. menu bar, header, pagination, ...). And for the last the public\_html/ folder is where all the accessible pages are, (i.e. index, contact us, …). below is a brief description of the most important files:

* try\_login.php: If there is no $\_SESSION variables (not connected), we look in the DB, and we connect it if there is a matching entry. If the guy was connected, we unset the variables;
* index.php: display blog posts and use 3 GET variables (username for displaying only its posts, ppp for the number of post per page, and page for displaying the right page);
* pagination.php will generate the right things according to these variables;
* aboutus.php: display a static page about us;
* contactus.php: js disabled version of the modal view (stored in includes);
* sendmail.php: get the input from contactus.php and send the email;
* new\_post.php: Form using CKeditor, that will be filled and sent;
* send\_post.php: Get the above inputs and send them to the DB;
* edit\_post.php: identical to new\_post.php except that the inputs are filled with the last message (matching the id given in the url) and does not use the same db functions (update instead of insert into);
* send\_edited\_post.php: same principle as send\_post.php;
* delete\_post.php: delete the post matching the id in the url;
* country\_select.php: list of countries;
* footer.php: simple footer of all the pages;
* head.php: function that print the all <head> with a variables in parameter (that will go in the <title>HERE</title>;
* header.php: header of the pages (Dillon Hall photo);
* include\_modals.php: register all the modals windows to be included in all pages;
* modal\_XXXXX: modal window that match the name (contact, login, logout, register);
* nav.php: According if the user is connected or not, display the nav\_user.php or the nav\_visitor.php;
* pagination.php only works on index.php (needs some variables of it) Display the number of pages;
* session\_start.php: start the session, and include the UserManager (used everywhere to connect / disconnect).

# Modules

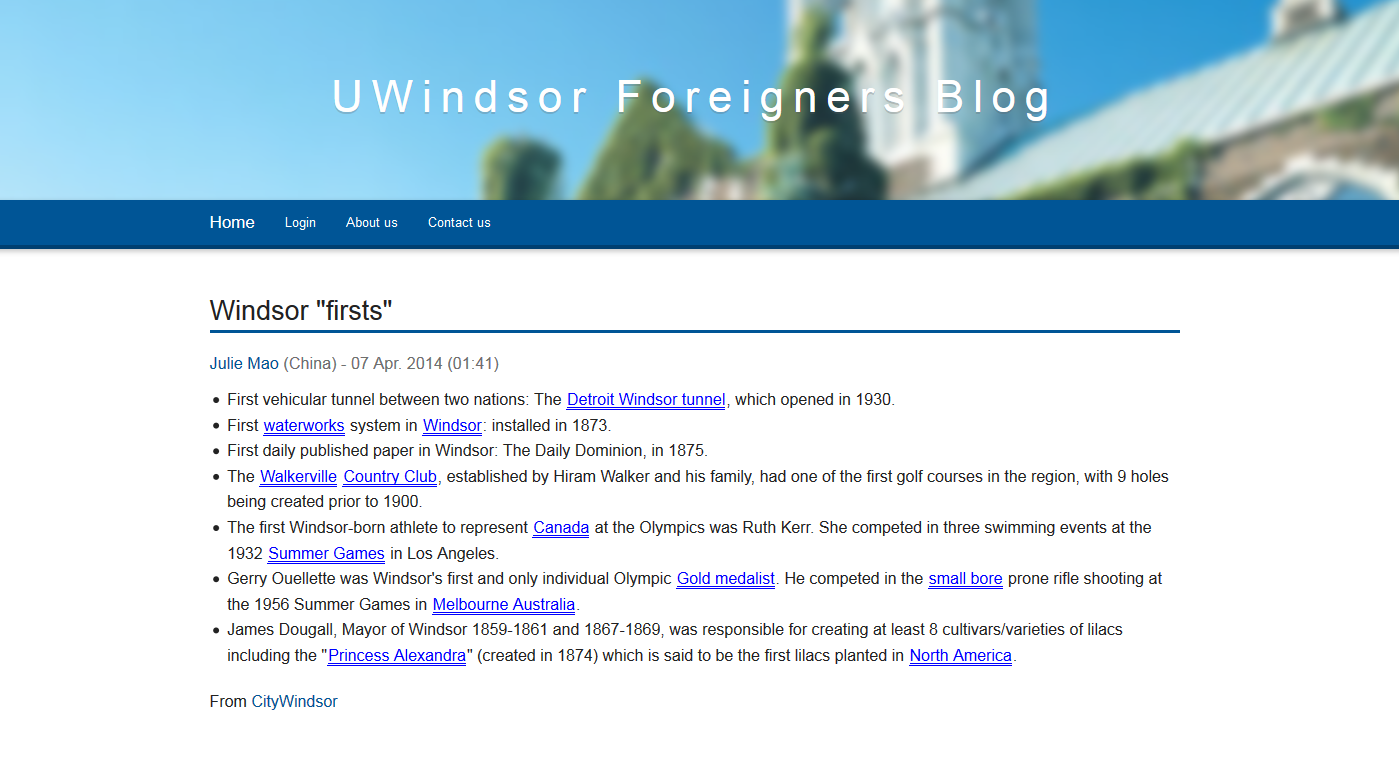


Figure 2 Main Page (Homepage)

In the Home Page the 5 most recent posts are presented, on the bottom has navigation between pages controls. Most of the modules are listed on this page, so let’s quickly go over them.

**Header module:** composed byhead.php and header.php.

**Menu module:** it is composed by nav.php, if the user is logged in uses nav\_user.php and if not nav\_visitor.php.

**Footer:** it is composed by footer.php.

**Pagination:** composed by pagination.php.

# Login

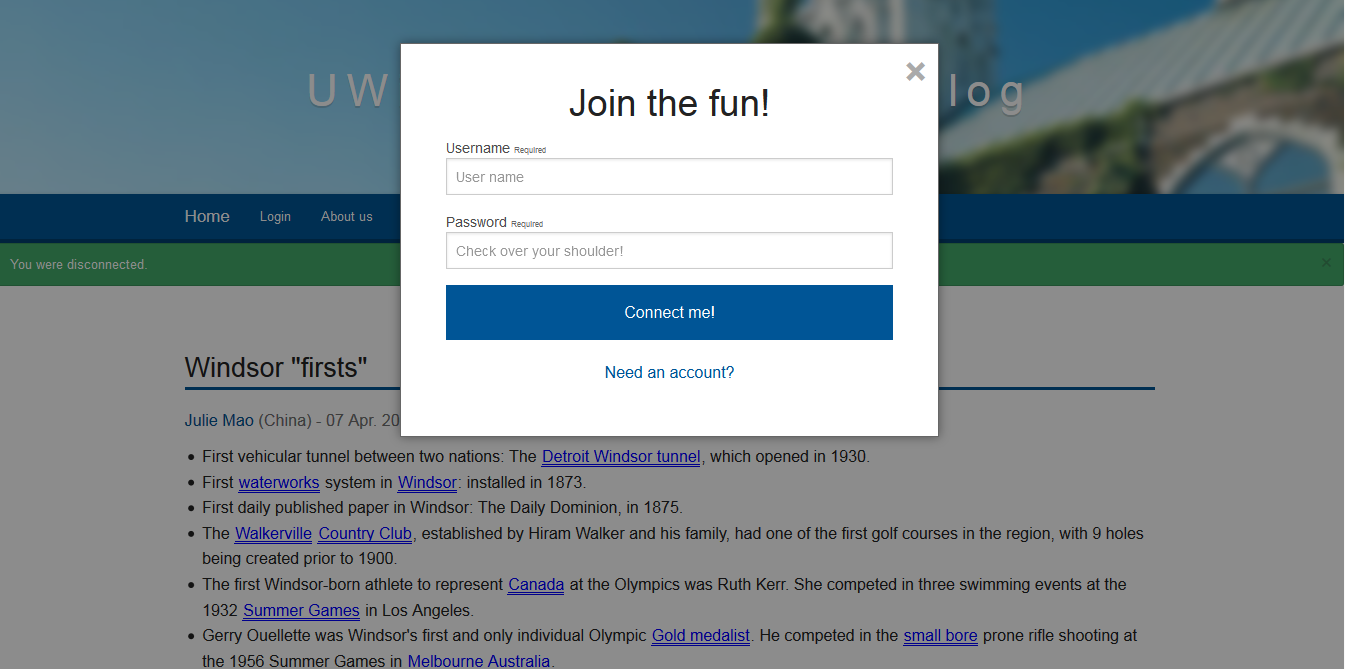


Figure 3 Login Frame

The process of login is divided in tree layers, modal\_login.php defines the appearance and form (figure 3) after click in connect me the data goes to try\_login.php that do security checks then call the UserManager.class.php to query in the database to check if the user and password are correct.

# Register

The register operates similarly to the login. The differences are that in the register the frame (Figure 4) is constructed by modal\_register.php which then calls register.php that check if the data is valid and sends this data to the UserManager.class.php perform actions in the database.

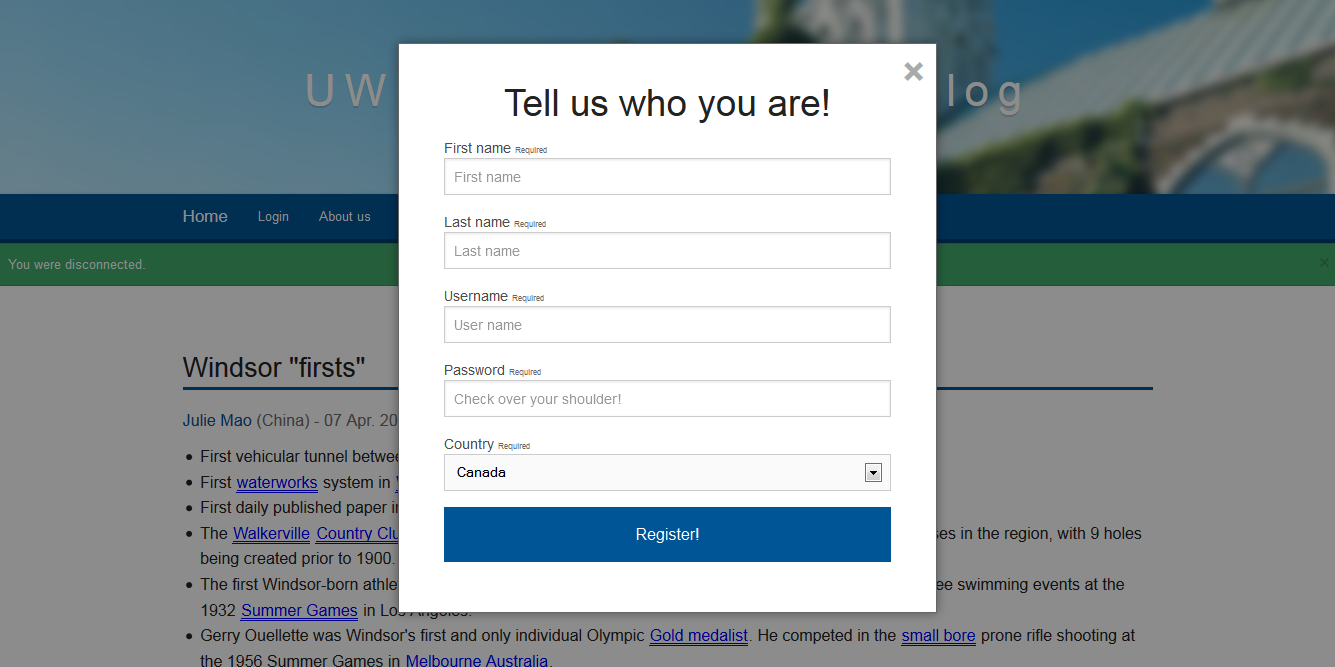


Figure 4 Register Frame

# New Post



Figure 5 New Post Page

The new post page (Figure 5) represents the file new\_post.php and is composed by two fields: Title and Text editor with a Body message. After a user typed his title and message and click on send the data are sent to send\_post.php that check the data for prevent attacks and validity. After this checking the data is sent to the PostManager.class.php that performs operations on the database. Can be presented to the user as javascript or html5.

# Edit Post

Using edit\_post.php instead of Works the as new post page but the fields title and body message are filled with the message that are being edited. Can be presented to the user as javascript or html5.

# Delete Post

Represents edit\_post.php and is responsible for the call to PostManager.class.php to delete from the database. Can be presented to the user as javascript or html5.

# About us

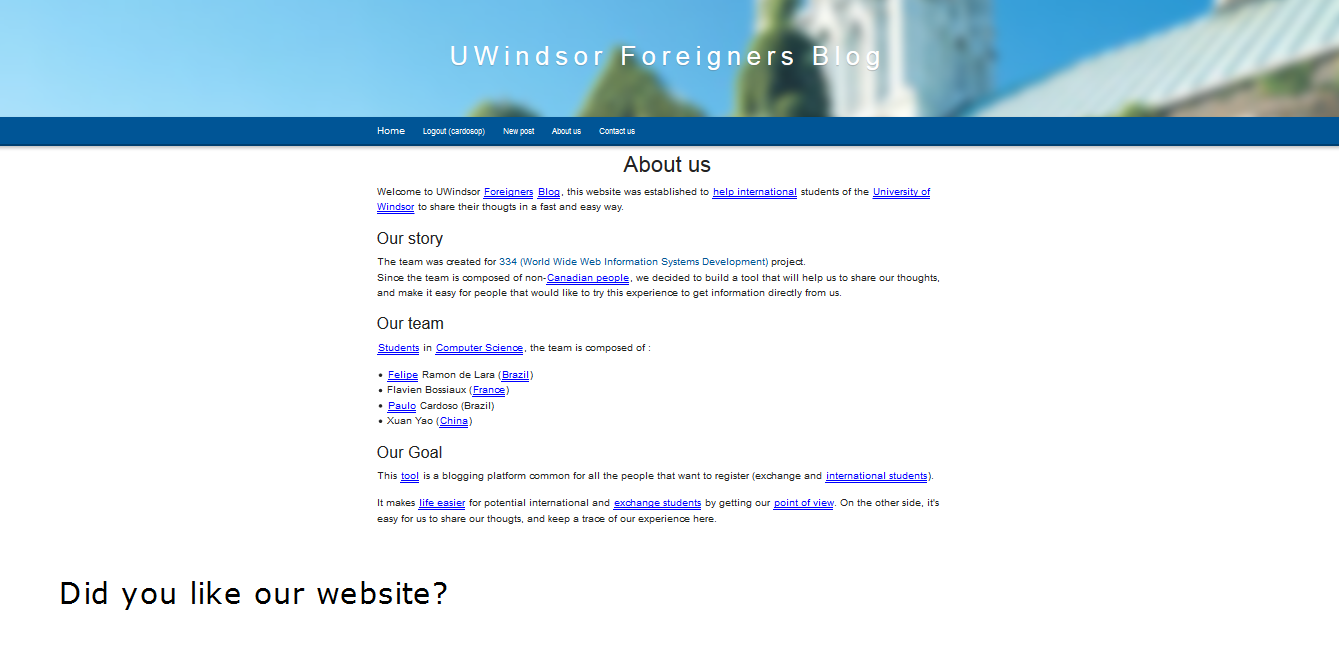
****

Figure 6 Page About Us

In the page about us (Figure 6) that represents the file aboutus.php. This page has a SVG element that is the text at the bottom “Did you like our website?”.

# Contact Us

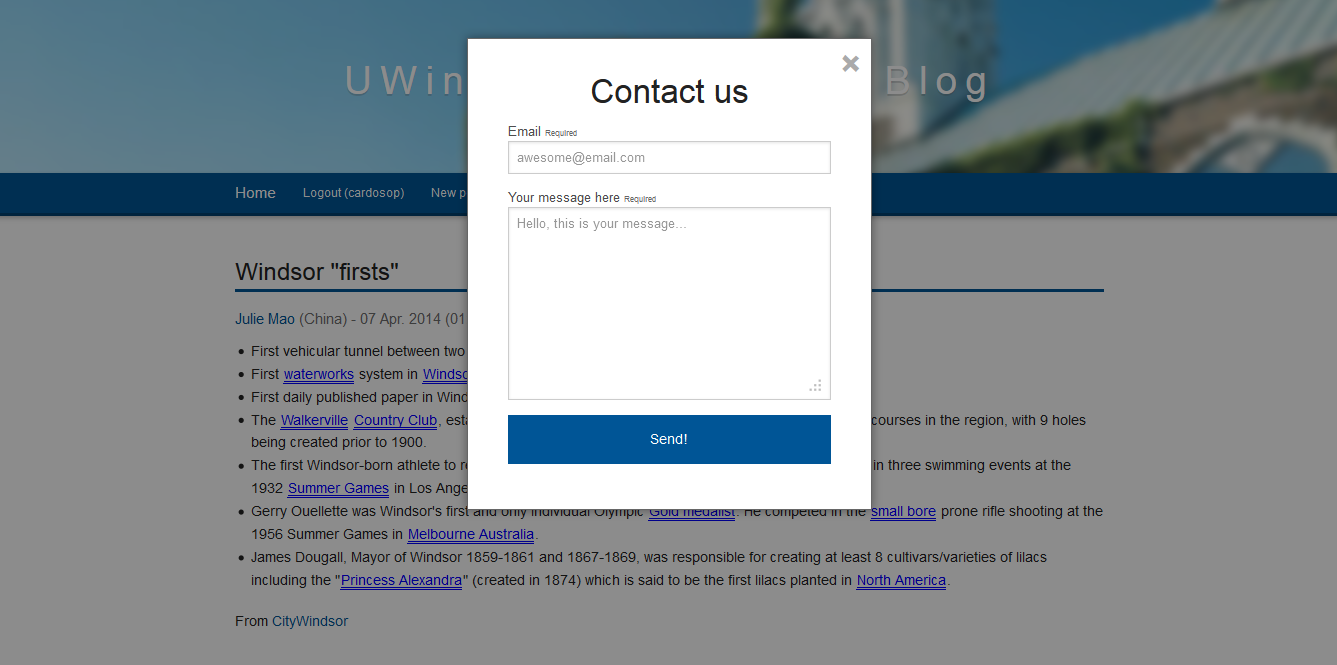


Figure 7 Contact Us Frame

The Contact us frame (Figure 7) works similarly to the login frame using a modal. The modal that defines the frame is modal\_contact.php that sent the data acquired to contactus.php that validate the data and calls send\_mail.php to send the email to the admin.