COMP 5531
Social Network Project

Marc Belley - 5701295
Michael Burkat - 5799236
Dmitri Grijalva - 9236880
Geoffrey Stanley - 5064961
Thomas Tigé - 9676007

1 Description

Our team's Private Online Social Network System (POSN) is named Laicos. Our primary goal was to turn Social around, back to simpler times, with a clean user interface and basic functionality that users can immediately understand. It is a fully functional self-contained social network, with user to user relations, and user to groups relations. Users can create and manage groups. Users can post content to walls, either to their own walls, other users, or to groups.

2 Assumptions

Design and implementation were done with a number of assumptions in mind:

- Implementation is to be done using the Code Igniter framework, on PHP with MySQL as a database. This was indicated by the requirements on the CrsMgr website.
- The site needs to be easily deployable to other servers in a small package, as described in the project requirements.
- Features would include as much as possible from the stated requirements, utilizing libraries when possible to minimise the amount of code that would have to be written.

3 Limitations

The limitations of our project were mainly related to management, time and knowledge issues.

- As members were unfamiliar with PHP and web design, and it was by no means a core part of the course, it required large amounts of learning for all members.
- Limits in quality and quantity of what we could fully implement, as time constraints of 11 weeks to design, implement and test a website with a large feature set are not realistic.
- It became clear throughout the development that a complete feature set would not be realistic. For example, things like resharing posts which would be a basic feature of a social network were not implemented to an adequate level.

The biggest impact on development that these limitations had for Laicos' development were that two non-functional requirements, security and stability, were not looked at in-depth. Realistically, Laicos is not ready for any production environment, however it could serve as the beginning of a social network framework.

4 Applications Supported

The Laicos platform supports a number of features, as recommended with the initial project requirements document. Our team attempted to strike a balance between time, feature breadth and feature depth.

By taking a top down approach, we started by figuring out the necessary parts and implementing those first. These were elements like the login system, profiles and walls. Not all applications are perfectly implemented, but the majority are functional.

- Registrations by invite-only
- Browse and Add friends
- Browse and Join Groups
- Post to Walls of Friends or Groups
- Add Comments on Posts
- Private Messaging between Friends
- Edit your Profile
- Public or Private Profiles (not fully implemented)
- History of Post and Ability to Delete Them
- Broadcast a Youtube or Youtube Live URL in order to Stream from your Profile

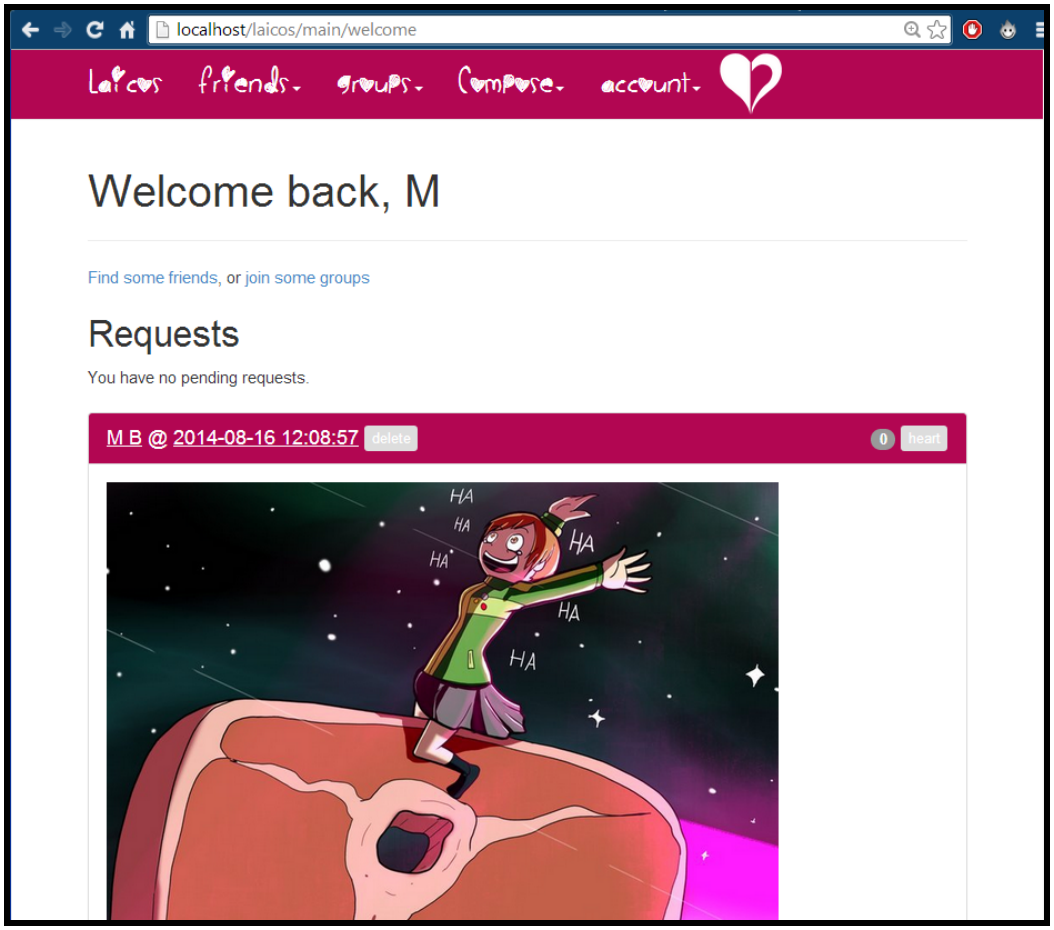
5 Interface Design

Overall Look & Feel

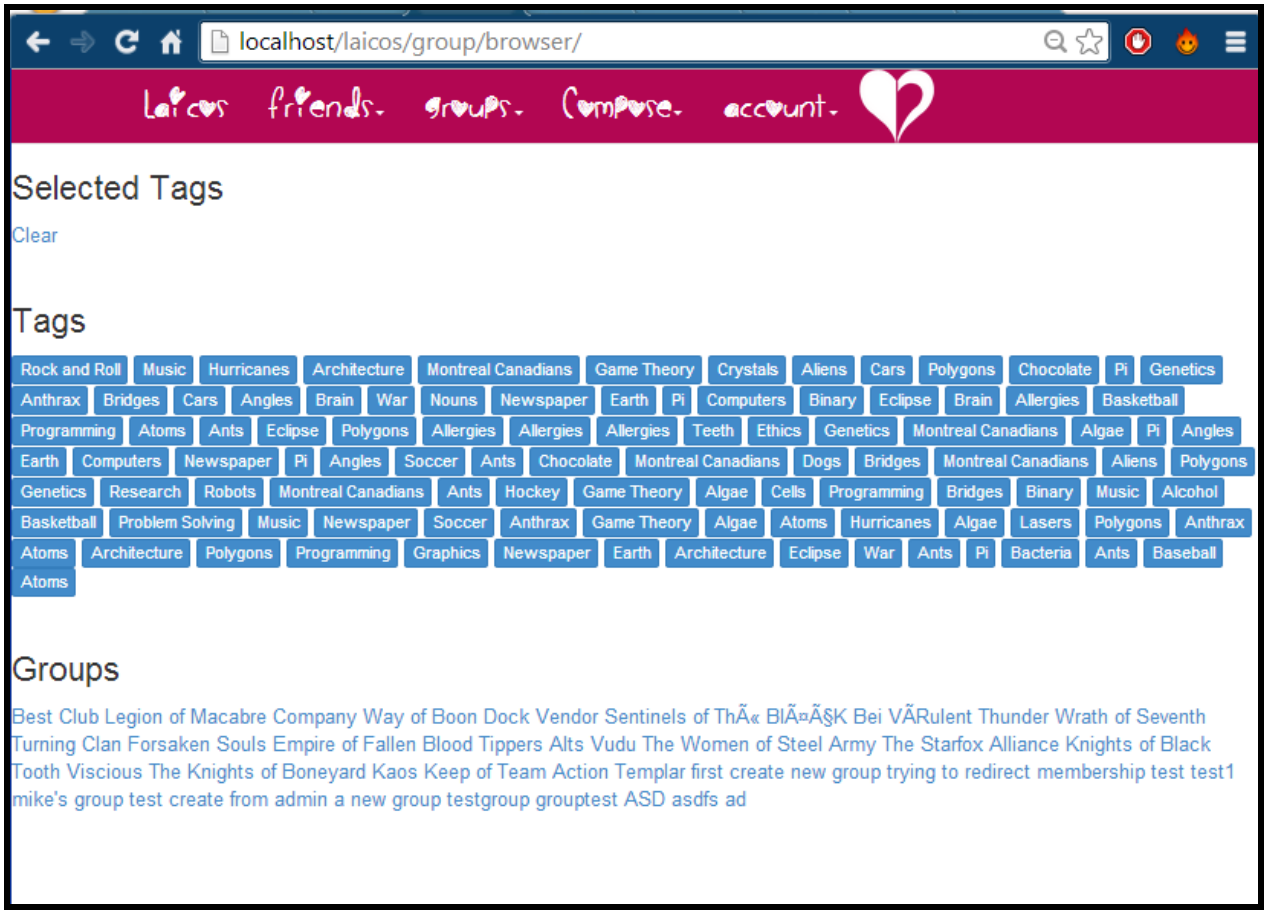
Our interface design was focused on simplicity and looking modern. The look-and-feel should be heart-related.

We decided to utilize the excellent Bootstrap framework for a lot of our CSS. The final outcome was aimed at simplicity and the 3-second rule. With the help of extensive UI testing, we wanted a user to log in and within 3 seconds understand the basics of our website. Furthermore, one of our goals was mobile friendliness, which we display in a number of pages, although due to time constraints we were not able to fix all the CSS appropriately.

Furthermore, some pages highlight understanding of server-client coding between PHP and Javascript. The final overall look and feel for browsing your main page:



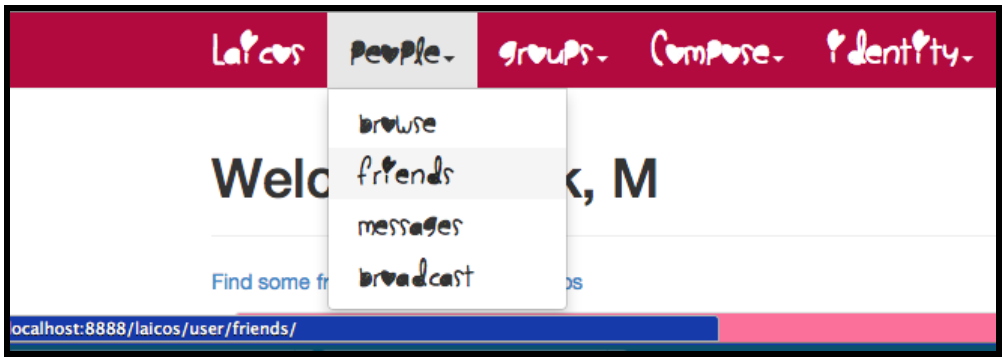
To join groups where you can read or post content:



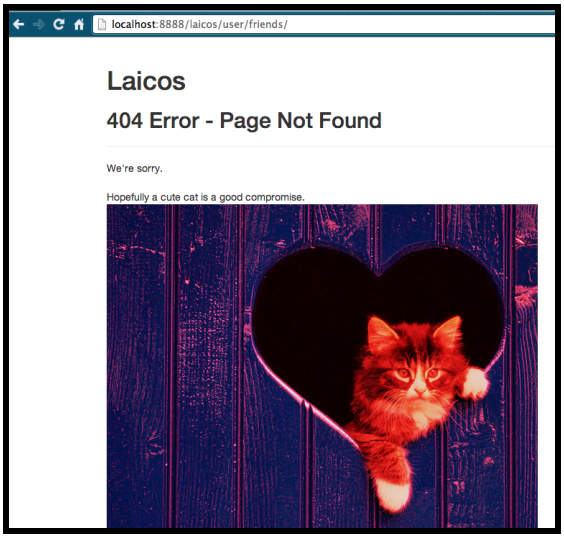
Menu Design

The menu design is the first element that highlights our wanted look and feel. It utilizes a custom font, downloaded by the client's browser, and has a number of menus with subitems.

To stay aligned with the goal of simplicity, the site was separated in people, groups, composition and identity sections. This could have been streamlined further, as people are in groups and you compose to both those elements, therefore clarity is not perfect as to what each menu has available.



No friends page… oops! However, a lolcat to help display graceful URL problems. In theory, this page would be different for logged in or logged out users and wouldn't really happen much anyway.

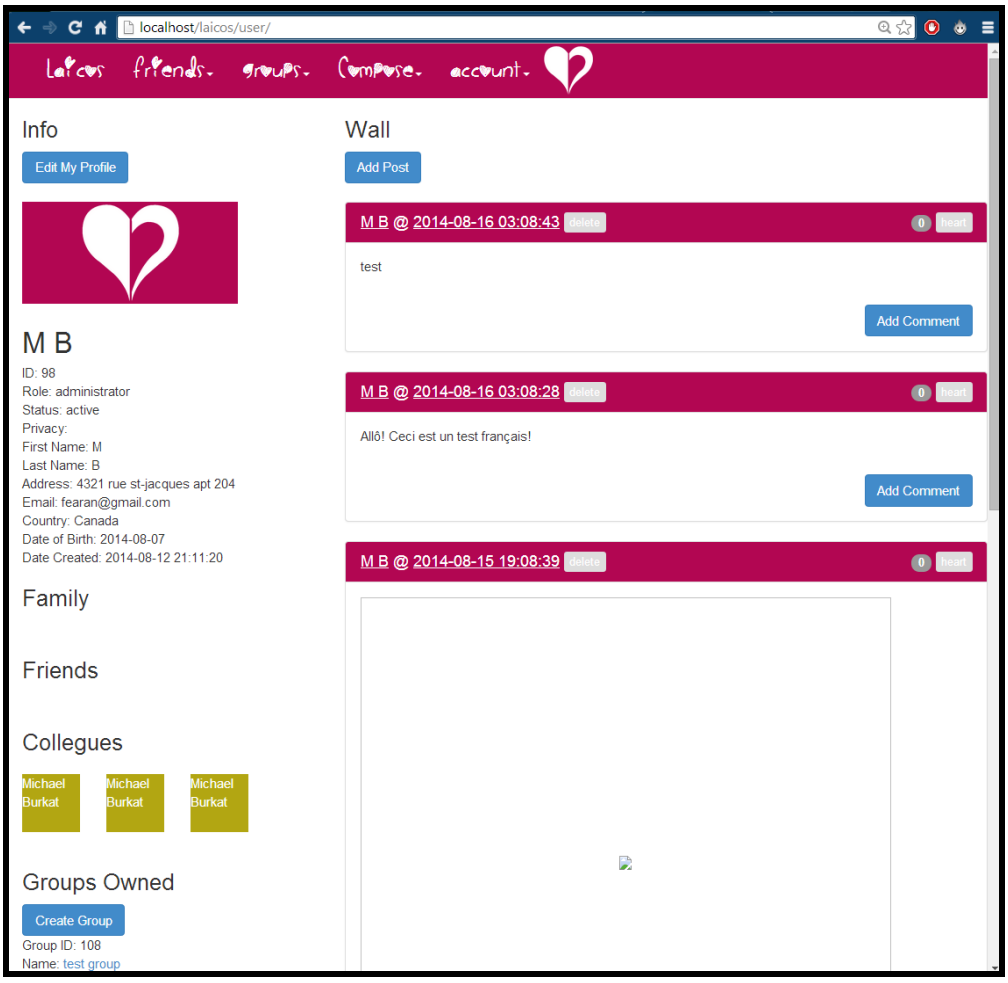


To see if anyone actively used the site, the project lead added in a trollish link in the menu bar. The friends page was never properly fixed, in order to have the funniest link to the 404 page.

Friend browsing and profile functionality was not affected, as a user's "friends" are already shown properly in their profile page. But we felt that the word should be included somewhere on the menu, to explain what is available on Laicos if you join a group or browse people.

User and Group Profiles

User and group profiles did not meet the highest levels of our standard standards, due to time constraints. The focus was on getting a functional and modern system in place, and we did not have the opportunity to clean up the CSS further and break down the profile information nicely.



However, from an implementation post of view, our focus was on building working functionality with as little bugs as possible, and from this perspective user profiles were a success. User and group display all the appropriate information properly.

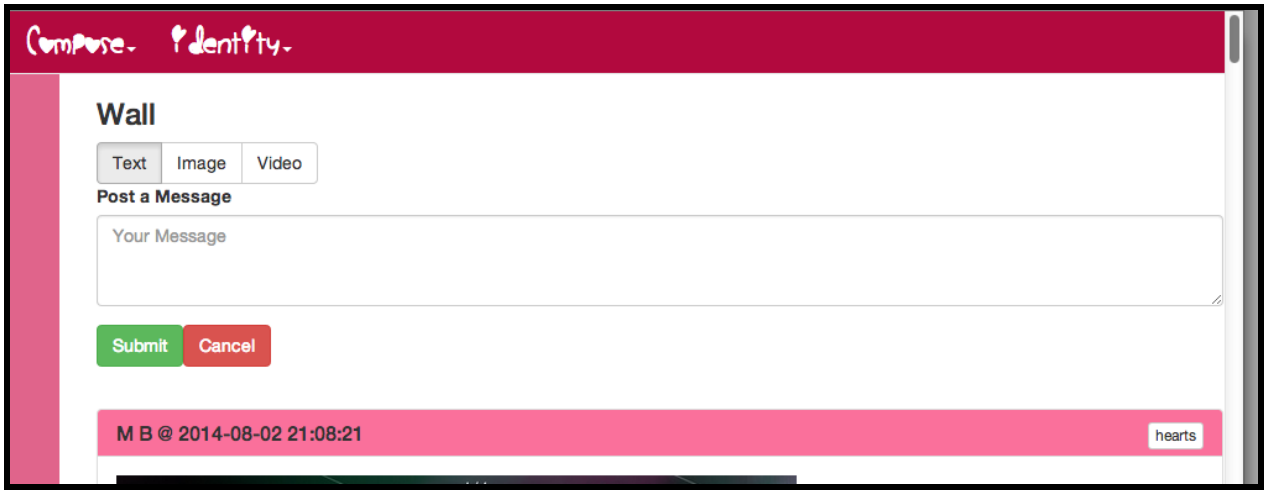
Furthermore, in order to display understanding of more advanced Javascript concepts, the User and Group profile pages were done with dynamic loading and posting of data.

Other views like post creation and private messaging show our use of modern CSS techniques to have a modern look and feel.

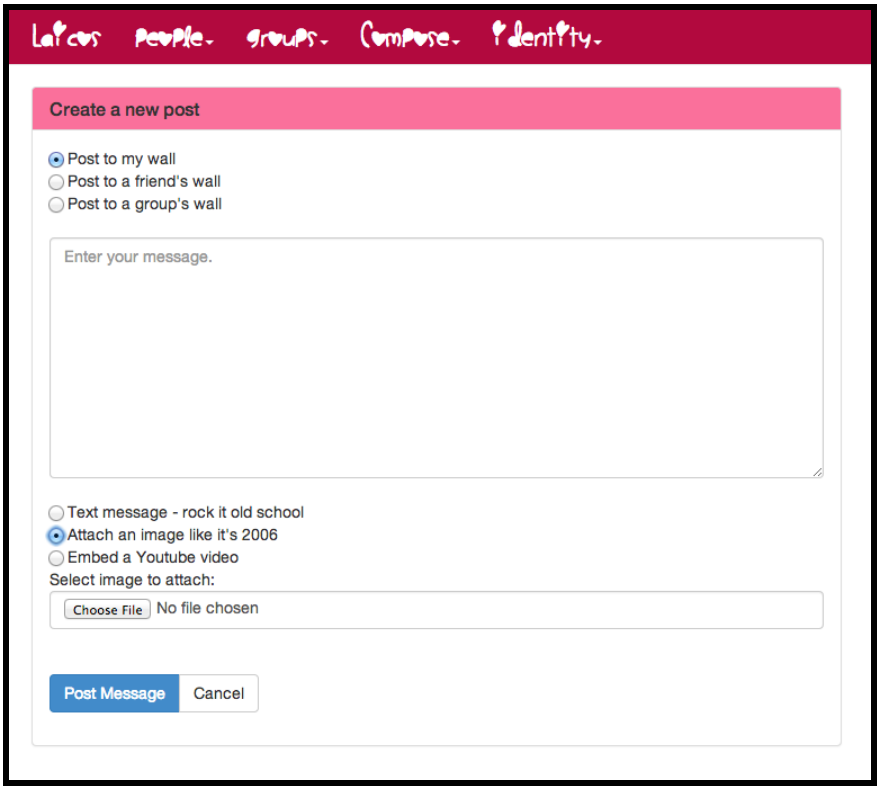
Creating Wall Posts

The posting interface can be accessed directly on a user or a group's wall. This is also where comments can be added. However, for simplicity of understanding, there is also a unified posting interface.

From a wall:

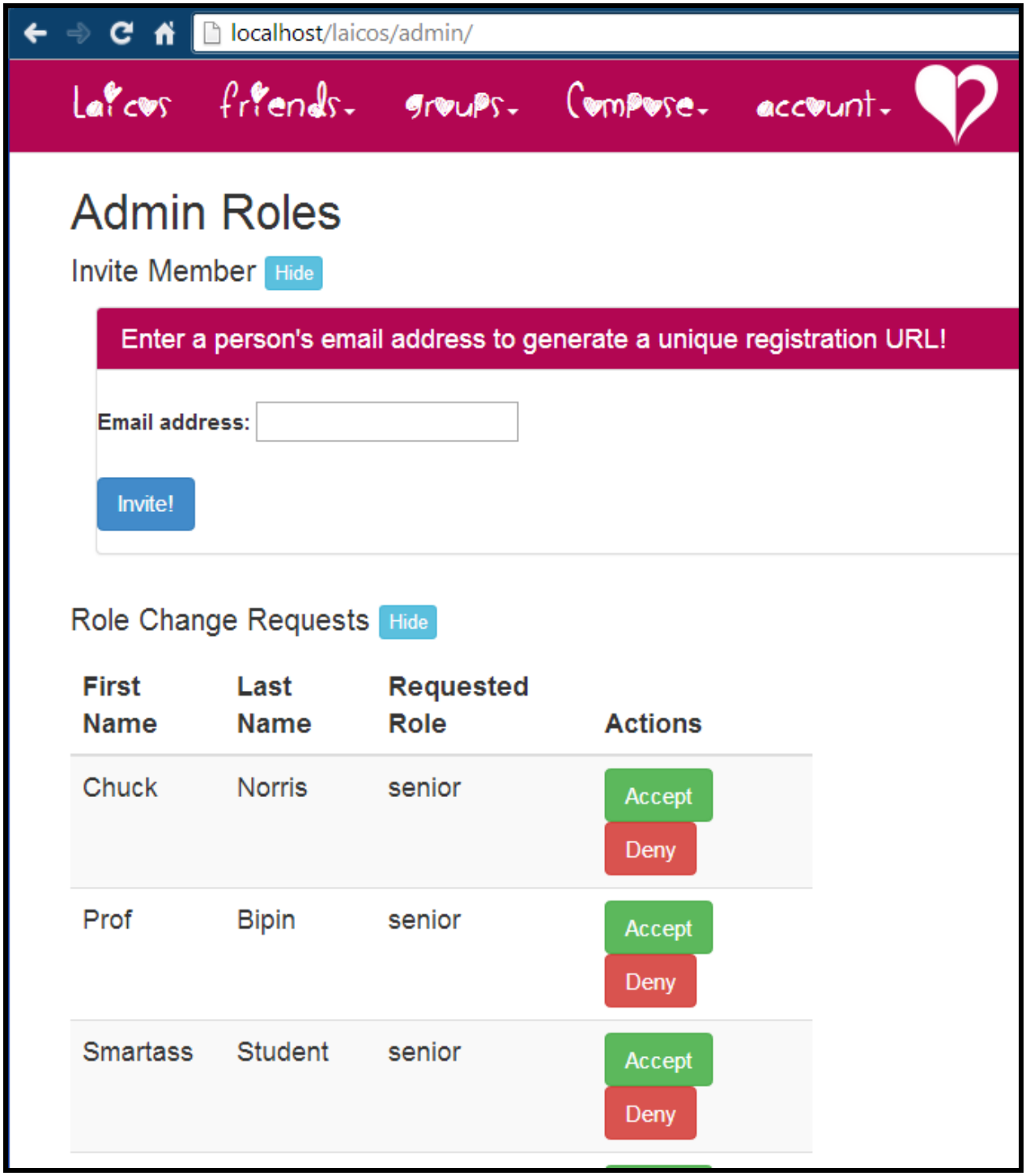


Unifiedposting interface:



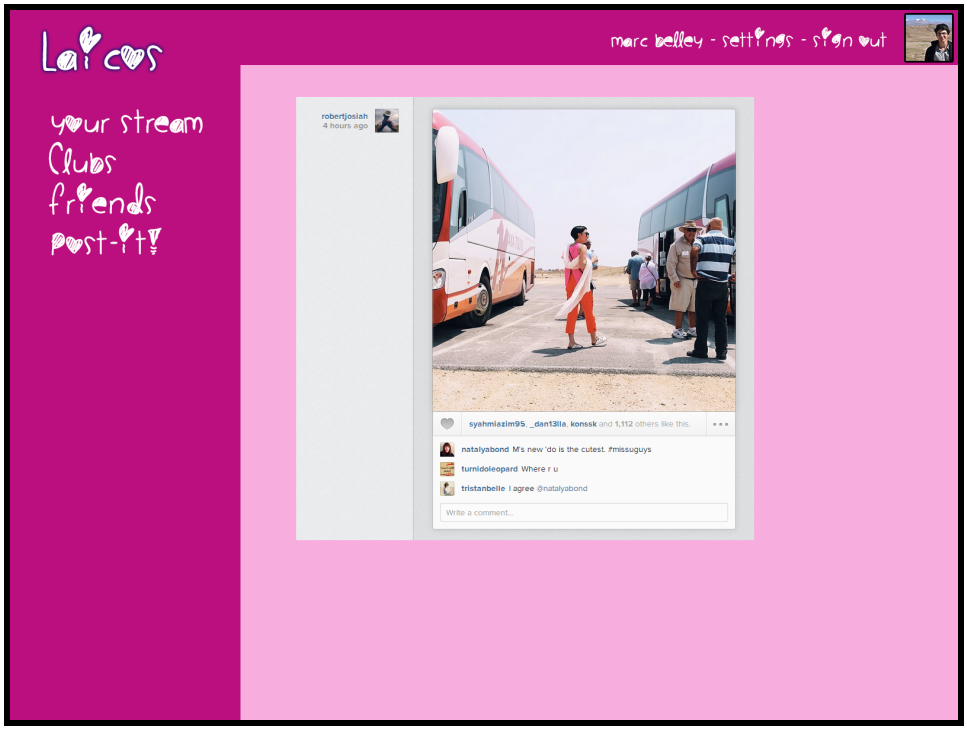
Admin Interface

It also proves remote access viability for full launch



Original Design Mockup

Earlier on in the semester, our team sat down together to discuss the overall look and feel we wanted, from an interface perspective and from a functionality perspective. This was done parallel to a quick mockup in Photoshop. The outcome was as follows:

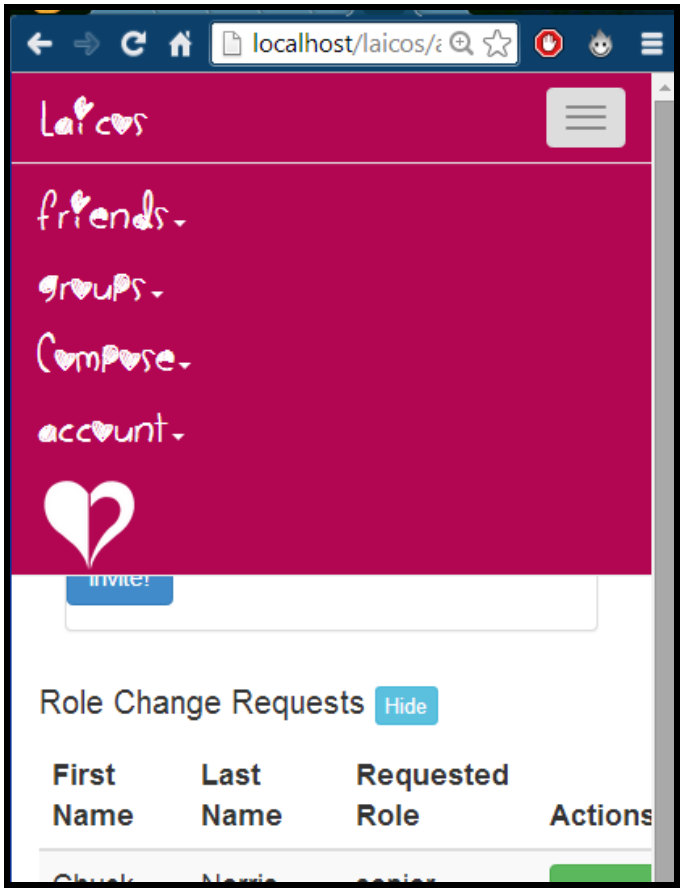


Our goal was to have a few, well functional features. The original concept centered around a few navigation features and a wall that would display posts. It was by no means our final design.

Our design concept, once implemented in PHP, was focused on easy to read text, clear navigation and simple, open white spaces. The basis for the design was “what would Google do?” Although their social platform, Google+, failed, the simple design concepts that it proposes are very interesting from a design perspective.

Mobile Friendliness

Laicos is mostly compatible with mobile browsers. With Bootstrap's help, everything is adaptable, including the menu which gracefully changes.



6 Model-View-Controller Breakdown

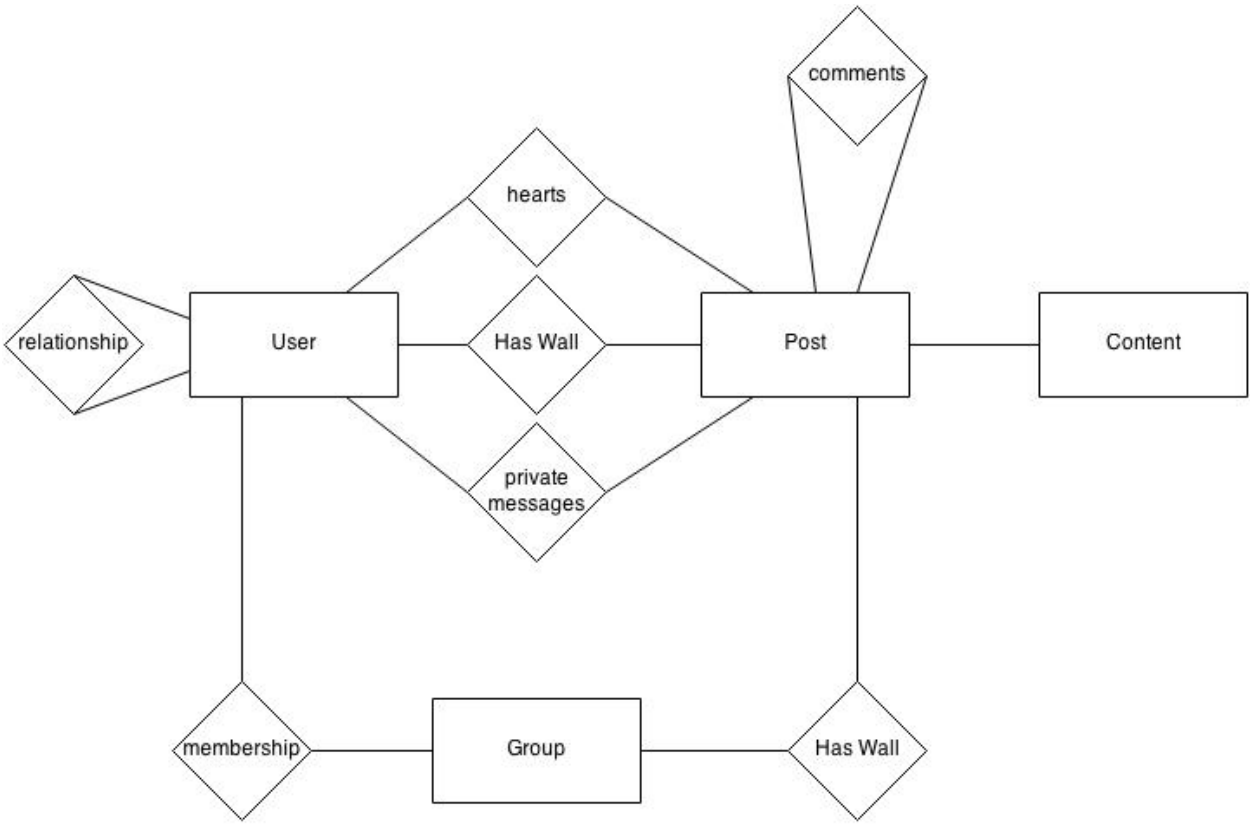
By using the Code Igniter framework, we achieved a better understanding of the classic MVC design pattern. It allows its users to easily make views, models or controllers.

The models are easily reusable pieces of code that represent individual classes in more object oriented programming. They contain most functions that can be reused.

The controllers represent the link with the outside internet. Each URL is dynamically generated page from the index.php file the framework starts with.

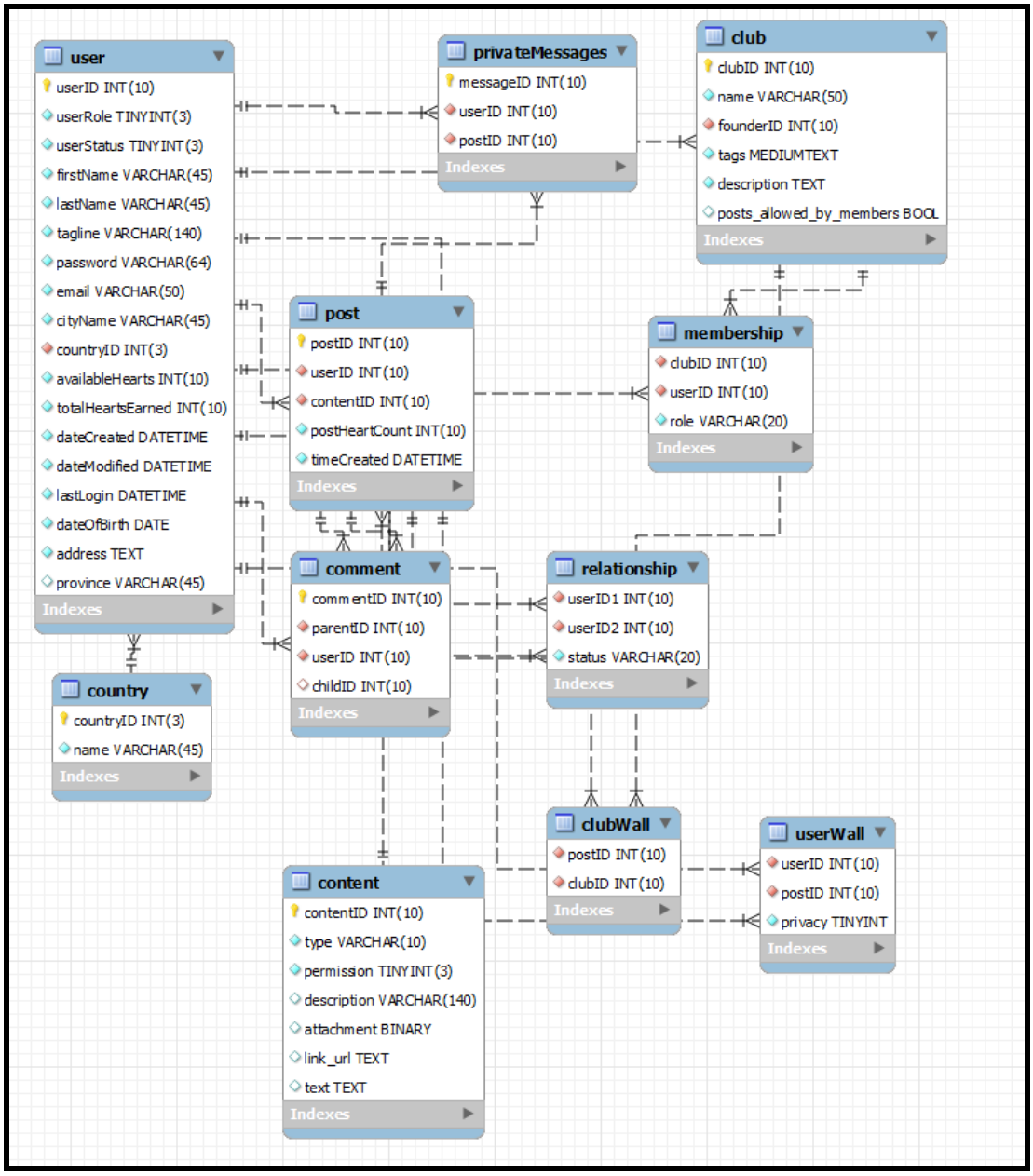
The views are the displayable HTML, CSS and JavaScript files.

7 E-R Diagram


comments
hearts
relationship
User
Has Wall
Post
Content
private
messages
membership
Group
Has Wall

8 Relational Database Design

Our database was based around users, groups, and posts. We wanted to keep things modular to easily add more features as required. For example, content is separate to a post, in case someone wants to reshare. We have a number of weak entities to define membership to groups and relationship between users.

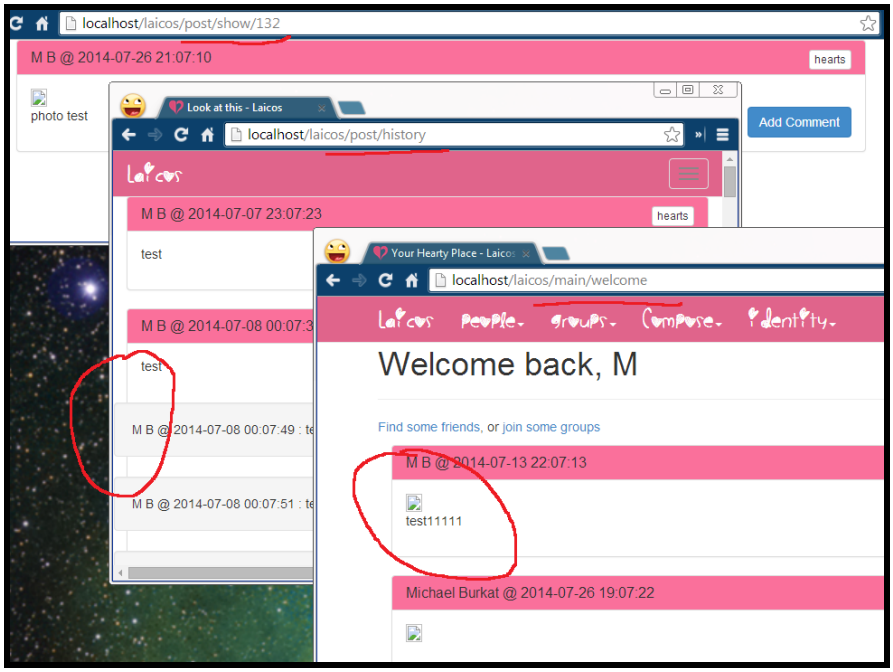


9 Team Member Responsibilities

Marc	● Manage timeline expectations, assigning tasks and issues ● Setup initial Code Igniter implementation, project repository ● Initial Bootstrap based CodeIgniter HTML-compliant pages ● Documentation (this document, readme file) ● Logo design and implementation
Michael	● User and group profiles with built in walls ● User and group browsers with filters ● Comments on posts ● Post system with images or videos to walls
Geoffrey	● Main welcome page ● Private messaging system ● Post display system, associated HTML and CSS pages
Dmitri	● Login & Registration ● Profile editing, password changing ● Gift system to give 4 different gift objects
Thomas	● Login & Registration ● Registration invite system ● Heart count on posts

Joint Responsibilities

- Review and comment on implemented features. Add any bugs to the issues list.



- Attend joint sessions to discuss project direction
- Keep track of issues assigned to them by project lead in Issues Log

10 Post Mortem: Project Performance

Key Accomplishments

For Example:

- Near complete feature implementation. Users can be invited, they can browse public groups and join private groups when they are properly invited. Senior users can moderate groups.
- Elegant user interface design that is simple yet powerful. The frontend uses well implemented JavaScript to make the pages dynamic.
- MVC design pattern used throughout

Key Problem Areas

[List problem areas experienced throughout the project. Be specific.]

For Example:

- Security risks were not properly covered, with holes such as SQL injection. This could have been resolved with parametrized SQL queries. A number of our queries are parametrized, but for complex SQL, Code Igniter's implementation is not great.
- There were a lot of features in the original requirements, which made the choice between well implemented and quickly implemented difficult.
- Mobile-friendliness was not considered as part of the design, which is one of the major lacunes of the requirements.

Risk Management

Project risks that have been mitigated:

- Coordination with non-standard schedules, which was mitigated with good source control behaviors.

Outstanding project risks that need to be managed:

- A number of small bugs that could have crept in. We did a lot of active Q&A with most systems, and applied agile processes, so this should have been mostly mitigated.

Overall Project Assessment

[Score/rank the overall project assessment according to the measures provided. A 10 indicates excellent, whereas a 1 indicates very poor.]

Criteria	Score
Performance against project goals/objectives	1 2 3 4 5 6 7 8 **9** 10
Performance against planned schedule	1 2 3 4 5 6 7 8 9 **10**
Performance against quality goals	1 2 3 4 5 6 7 8 9 **10**
Performance against planned budget	1 2 3 4 5 6 7 8 9 **10**
Adherence to scope	1 2 3 4 5 6 7 8 9 **10**
Project planning	1 2 3 4 5 6 7 8 **9** 10
Resource management	1 2 3 4 5 6 7 8 **9** 10

Project management	1 2 3 4 5 6 7 8 9 **10**
Development	1 2 3 4 5 6 7 8 **9** 10
Communication	1 2 3 4 5 6 7 8 9 **10**
Team cooperation	1 2 3 4 5 6 7 8 **9** 10
Project deliverable(s)	1 2 3 4 5 6 7 8 9 **10**

Additional Comments:

Laicos turned into a great and interesting project.

11 Post Mortem: Key Lessons Learned

… to be added from post mortem doc

https://docs.google.com/document/d/191citT_Ch9FS_rxRrX5tE1_a_COxycBwEWcGMpaSP2k/edit

12 Detailed Commit Log