openjournal.

Andrew Smith

Final Report

Project Concept Proposal

Purpose:
Content-management systems (CMS) are arguably some of the most important web applications is use today. Most large websites use some form of a CMS to manage their content in a meaningful way. The main focus of openjournal is to provide the same tools and features as many large-scale CMSs, but in a much more simple and minimalistic way. In addition, the system will also have a developer-friendly mentaility, which will be vital to the creation of themes.

Context:
There are many content-management systems in use all over the world-wide web. One of the large probems with most CMS's is that you still need tech-savvy users to manage the website. This CMS will fit by focusing more on being user-friendly than powerful. This application will also be open-source and easily extensible, so that additional functionality can be added as needed.

Goals:
Many web applications strive to be user-friendly, but many content-management systems still have not adopted this practice. The general goals of this application are to both make it easy for users to manage websites and allow them to make changes to their website which would typically require expensive developers. We will reach this goal by providing an easy-to-use interface, as well as many web features that can be added to a website with little to no technical work.

Audience:
This application is expected to have a relatively diverse audience. The largest section of this audience will likely be small business owners who do not have the funds to hire full-time web designers to manage their websites. Another section will account for designers who are not also web developers. This application will give them both the ability to be as flexible as possible with their website according to their skill-level.

Functionality:
At its core this system will allow users to write and manage the content posted on their website. In addition to the core functionality, the application will focus on providing commonly used features which can be added to a user's website as desired. Exampes of these features would be: contact forms, photo galleries, social networking and FAQs.

Milieu:
The most popular similar projects are WordPress, Joomla! and Drupal. All of which are very exceptional CMS's, but they also have a steep learning curve. Some even require specialized training courses to become skilled in their use.

Novelty:
openjournal will differ from the above systems in a lot of ways, the largest of which is the focus on usability rather than power. It is the view of the developers that if more users can feel comfortable with your system, then it will be a more efficient system.

Resources: #Modified 09/19/2011

PHP 5.3.8: http://www.php.net/ Handles database intereractions and modification.	MySQL 5.5 http://dev.mysql.com/ The application will be database-driven, using MySQL.	HTML5, CSS3, Javascript http://www.w3.org/ Used for front-end interface of the application.	TinyMVC Framework - Andrew Smith Will be used to reduce low-level database interactions.

Challenges:
The most difficult portions of this project will be in design rather than implementation. There will be many hard decisions to make concerning usability. While usability can be considered a science, the usability of CMS system does not have a large following.

Measures:
This project will be considered successful when, after copious testing, it has been proven easy to use by less technical users. The evidence will need to show that the system aids small business owners manage their websites. It is very important that it will be easy for designers to create simple themes for this system.

Future Extensions:
Most extensions will come rather naturally after this semester comes to a close. As new features become requested by users, they will be added. For instance, different types of photo galleries, calendar applications, etc, will all be welcome additions to the applications after it's base code has been written.

Inspiration:

Motivation:
As a computer science student at an institution which emphasizes service, I've worked a lot with people who do not feel comfortable with computers. I have also felt the steep learning curves of similar projects. These things inspire me to create something which will be of much more help.

Profession:
I am, by trade, a designer and developer, so this project will allow me to exercise both my design and development skillset in an academic setting. This will be one of the largest projects I've ever worked on, as such it will make me more comfortable tackling larger projects.

Vision

The designer of this application has a firm belief that there is a need for more user-friendly, extensible and flexible content-management systems. CMSs are managing thousands of websites on the net, but very few of them focus on usability. The vision of this application is to fit into this usability niche, as well as being very handy and simple for designers.

Scope

Although the goal of this project is to keep it simple and provide a very flexible tool to small-business owners, the actual development of this sort of system is not simple in any sense of the word. This project does not intend to compete with CMSs that have been in development for years, but it does aim to bring a refreshing and simpler entry point to CMS applications in general. At the end of this development cycle, the project will be a user-friendly content-management system. As the word implies, the project will at bare-minimum be able to manage web content. All extra functionality is to be added as time permits.

Software Requirements Specification #Modified 09/19/2011

The focus of this section of the project report is to describe in more detail the applications functionality. Below, requirements of the application will be listed in terms of priority and prerequisites. As this project is very large in it's design, this document will be split into sections, with software requirements for each.

Content Management

1. **Dynamically store & retrieve content**
   Evaluation Method: Pages can be stored in a database and retrieved by application to be displayed to the user.
   Dependencies: (independent of application) database, server for testing
   Priority: High
2. **Allow editing of content**
   Evaluation Method: Dynamically stored pages can be edited by end-user
   Dependencies: 1. Dynamically store & retrieve content
   Priority: High
   Expected addition: WYSIWYG editing
3. **Arrange stored content into categories, for editing and navigational purposes**
   Evaluation Method: Pages can be filtered by their category
   Dependencies: 1. Dynamically store & retrieve content
   Priority: High
4. **Manage site navigation**
   Evaluation Method: User will be able to choose how site viewers navigate the site
   Dependencies: 1. Dynamically store & retrieve content, 2. Arrange stored content into categories
   Priority: High
5. **Manage media**
   Evaluation Method: User will be able to upload and post varying types of media.
   Dependencies: 1. Dynamically store & retrieve conten
   Priority: High

Manage Site Appearance

1. **Allow user to choose from predesigned themes**
   Evaluation Method: User can choose themes from an administration dashboard
   Dependencies: all content management
   Priority: High
2. **Create simple templating scheme**
   Evaluation Method: Designers should be able to create themes for this CMS with little to no knowledge of server-side code.
   Dependencies: 1. Allow user to select themes, all content management
   Priority: High-medium
3. **Ability to modify administration theme**
   Evaluation Method: Users can customize the look and feel of the back-end application
   Dependencies: 1. Dynamically store & retrieve content
   Priority: Medium

Application Settings

1. **Manage users**
   Evaluation Method: Administrators can add/modify users for the system
   Dependencies: none
   Priority: High
2. **User priveleges**
   Evaluation Method: Admins can create users with varying degrees of access to content management
   Dependencies: 1. Manage users
   Priority: Medium
3. **Site maintenance mode**
   Evaluation Method: Admins can take the site down temporarily to make changes
   Dependencies: none
   Priority: Low

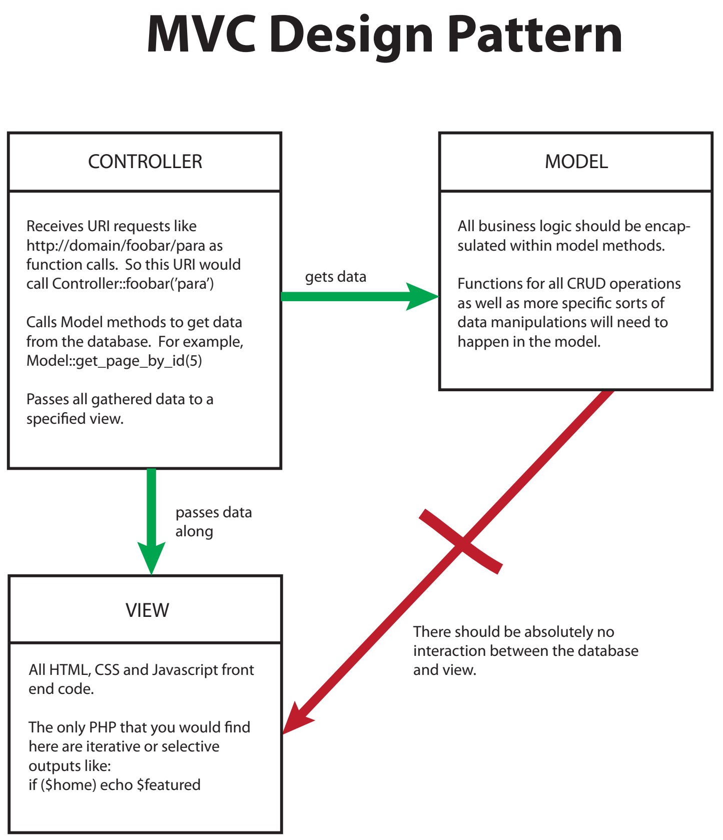
Modules

1. **Provide set of “modules” users can add to site for additional functionality**
   Evaluation Method: Users can add things like precreated contact forms/jquery sliders.
   Dependencies: All content-management and appearance
   Priority: As time permits

Software Design and Architecture

MVC design pattern

Before delving deeply into the design and architecture of openjournal. It is important to understand how it is constructed. openjournal uses a Model-View-Controller (MVC) design pattern, which is very common in web development today.


MVC Design Pattern
CONTROLLER
Receives URI requests like http://domain/foobar/para as function calls. So this URI would call Controller::foobar('para')
Calls Model methods to get data from the database. For example, Model::get_page_by_id(5)
Passes all gathered data to a specified view.
gets data
MODEL
All business logic should be encapsulated within model methods.
Functions for all CRUD operations as well as more specific sorts of data manipulations will need to happen in the model.
passes data along
VIEW
All HTML, CSS and Javascript front end code.
The only PHP that you would find here are iterative or selective outputs like: if ($home) echo $featured
There should be absolutely no interaction between the database and view.

openjournal system functionality

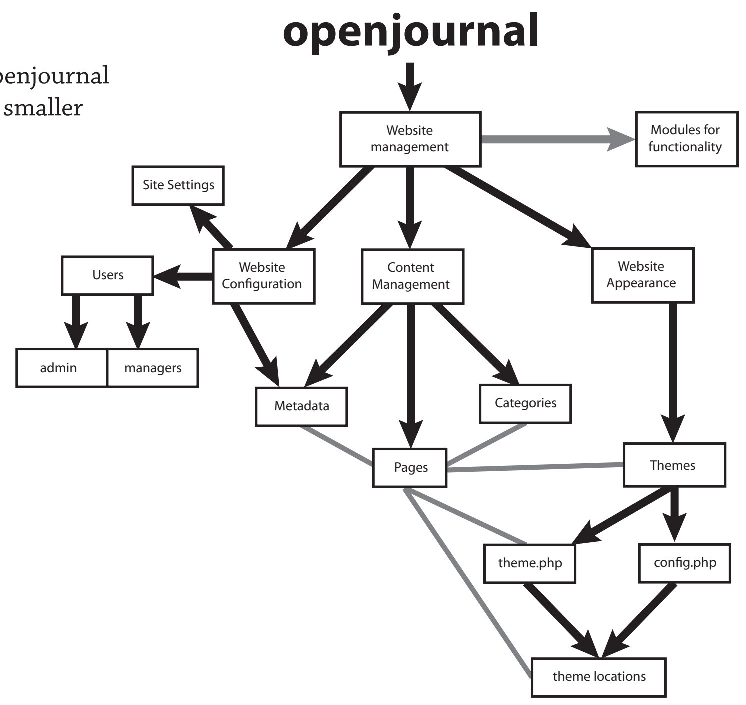
To the right you'll find a model describing the functionality of openjournal at a high level. This model starts breaking down the system into smaller more digestable pieces in a relatively simple flowchart.

I would like to note a few things here.

For the sake of simplicity I have intentionally left some of the functionality out of this model.

Next, you'll see the grey arrow pointing to modules for extra functionality. This is grey because this part of the project will be completed as time permits.

Grey edges connecting nodes mean that they depend on each other. For example, "Pages" depend on "Themes" and "Categories".



openjournal object map

Below is an illustration of all of the objects used by openjournal, and how they relate to each other in terms of inheritance and usage. openjournal is a relatively large system with diverse and modular functionality, so it is important to understand how these objects are used.

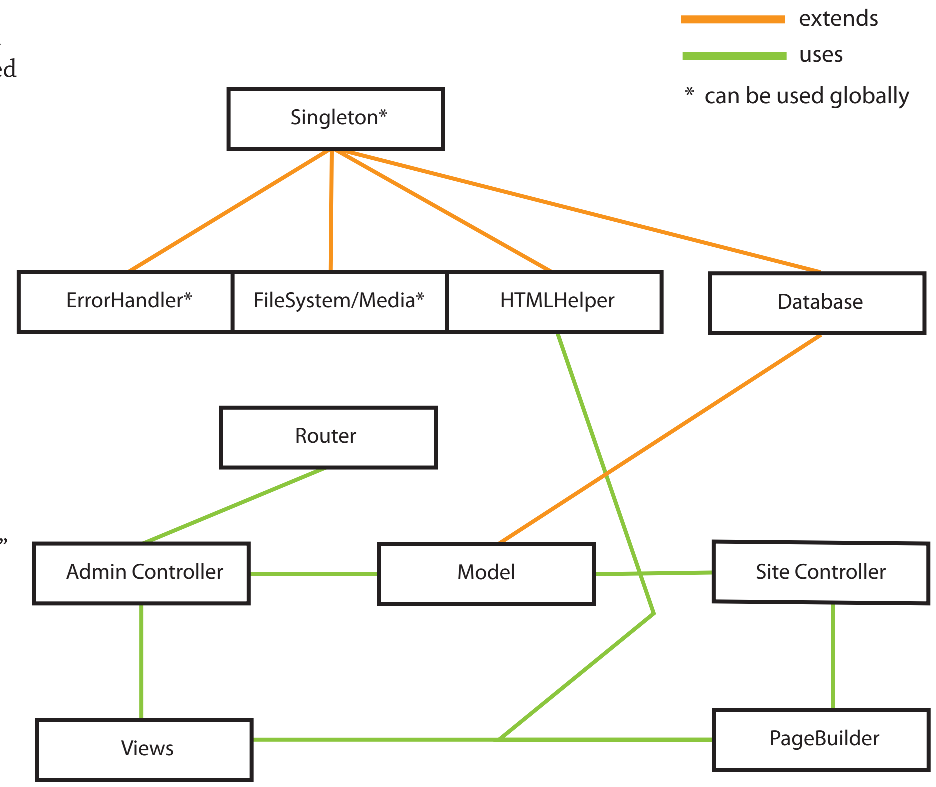
For the most part, the objects to the right are self-explanatory, but I would like to note a few specific things about some of the objects.

The “Singleton” object (or Singleton design pattern), is used for objects which only need to be instantiated once througout a large system. This is important for many of these objects, in particular the “Database” object needs to have the same instance system wide.

The “Router” object is used to map URI requests to correct Controller functions. For example, if http://domain/foobar is requested, the Router will call the “foobar” method of the controller.

There are two controllers, one for Admin and one for the general Site. They have distinct functionality. The “Site Controller” manages all of the things a typical user would see when browsing an openjournal website (pages, categories, etc). While the “Admin Controller” manages all administrative functions in the admin interface. This is very modular, and it could easily be the case that someone wants to use the openjournal system, but completely change the back-end (administrative) interface.

Objects and inheritance



Directory structure

To the right is the proposed directory structure for the system. Changes are likely, as new functionality will be added in the modular way of adding new views/modules/themes/etc.

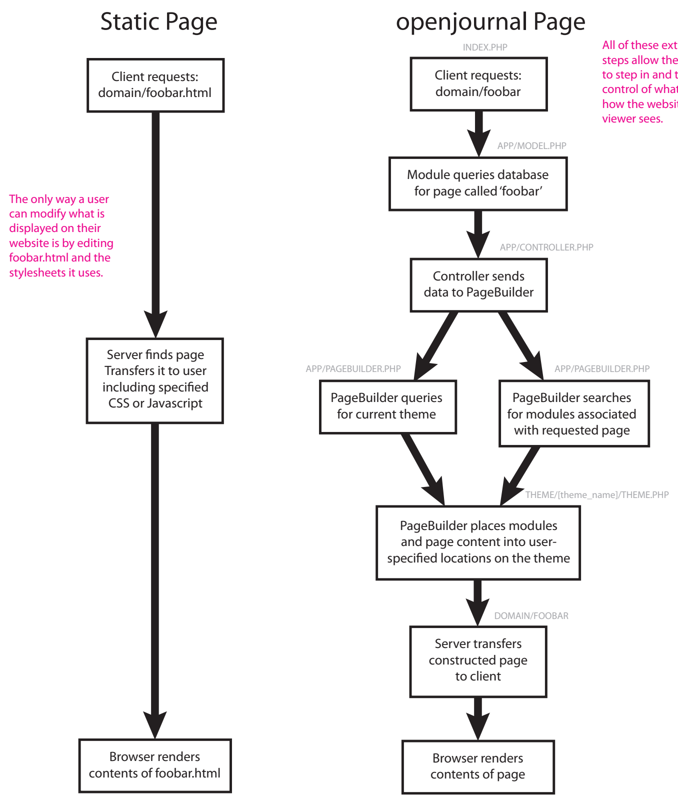
Directory Structure

```
openjournal/
    app/
        libraries/
            errorhandler.php
            htmlhelper.php
            filesystem.php
            database.php
            singleton.php
        admin/
            css/ ...
            js/ ...
            img/ ...
            views/
                add_form.php ...
            admin-controller.php
        controller.php
        model.php
        pagebuilder.php
    themes/
        default/ ...
            theme.php
            config.php
            theme.css
    modules/
        image-slider/ ...
            config.php
            template.php
    media/
        img/
        doc/
    index.php
    router.php
```

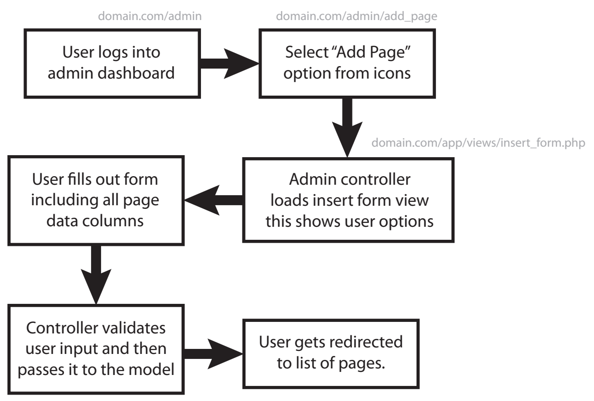
Typical admin workflow

Here are a few workflow models of how a user would go about accessing openjournal pages, and how an administrator would add new pages.

How pages are retrieved

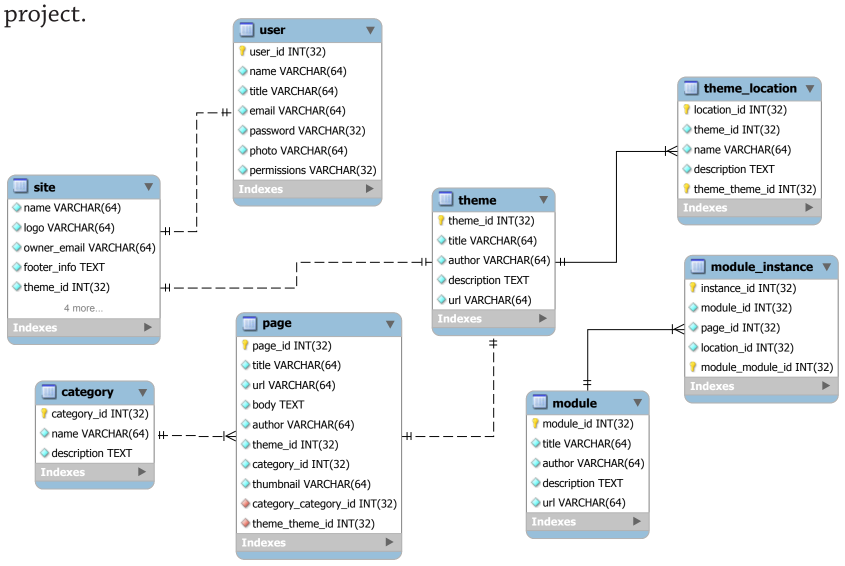

Static Page
Client requests: domain/foobar.html
The only way a user can modify what is displayed on their website is by editing foobar.html and the stylesheets it uses.
Server finds page Transfers it to user including specified CSS or Javascript
Browser renders contents of foobar.html
openjournal Page
INDEX.PHP
Client requests: domain/foobar
All of these extra steps allow the user to step in and take control of what and how the website viewer sees.
APP/MODEL.PHP
Module queries database for page called 'foobar'
APP/CONTROLLER.PHP
Controller sends data to PageBuilder
APP/PAGEBUILDER.PHP
PageBuilder queries for current theme
APP/PAGEBUILDER.PHP
PageBuilder searches for modules associated with requested page
THEME/[theme_name]/THEME.PHP
PageBuilder places modules and page content into user-specified locations on the theme
DOMAIN/FOOBAR
Server transfers constructed page to client
Browser renders contents of page

Adding pages


domain.com/admin
User logs into admin dashboard
domain.com/admin/add_page
Select "Add Page" option from icons
domain.com/app/views/insert_form.php
Admin controller loads insert form view this shows user options
User fills out form including all page data columns
Controller validates user input and then passes it to the model
User gets redirected to list of pages.

Database ER diagram

An ER diagram shows the relationships between the tables of a database. As this project is a a very large database-driven website, it is important to see these relationships. Good database design can save you massive amounts of time in the development of the back-end of a web development project.



Implementation Details #Modified 10/15/2011

With a system as large as this, may not be useful to explain each file, as many of the files introduce only small bits of extra functionality (e.g. stylesheets), so in this section I will be explaining the important files which are required for this system to work.

- **index.php**
  The only purpose of this file is to set a few important global variables pertaining to site configuration, and then yield control to the router.php file.
- **app/router.php**
  This may be the most important file in the entire system. At a high-level, this file includes all relevant libraries and classes that will be used throughout the system. Next, it will turn a user HTTP request (http://localhost/openjournal/foo/bar) into a controller function call. So with the example above, the system will call Controller::foo(bar). It is a difficult idea to grasp, but it's extremely powerful because it means that you do not have to create new files for additional functionality.
- **app/controller.php**
  This file is the brain of the application. It holds the Controller class, which has methods which function as a traditional web application's PHP files. Instead of add_record.php, you'll have Controller::add_record(). These functions interact with the Models to get data, and then uses the data in relevant ways.
- **app/model.php**
  This file holds the Model class, which is an extension of the Database class. This file holds all business=logic used in querying the database for useful information. It passes this information to the controller.
- **app/libraries/database.php**
  This file holds the Database class, which is a huge abstraction from traditional database queries. This allows you to, instead of writing long queries, write nice little functions like Database::insert('page', $columns)
- **app/libraries/htmlhelper.php**
  Helper functionality for Views. The most prominant features here are a create_table, create_insert_form, create_edit_form and create_list functions.
- **app/admin/admin_controller.php**
  This is a second controller file. It has a specific controller for the Admin section, which has tons of methods including add_page, edit_page, change_theme, dashboard.
- **themes/ directory**
  Contains default and user themes.

There are many more files to this system, which are not mentioned above. These are generally self-explanatory, and accompanying inline and header-style documentation to explain their functionality.

Known Bugs and Other Issues – in openjournal 0.1.15 alpha #Updated 12/12/2011

Due to the size of this project, the current status of functionality is not yet at alpha or beta status. Some functionality has yet to be created, but all of the base code has been written, and new features are being added every day.

Missing functionality #Updated 12/11/2011

- Several functions listed in dash are currently not implemented
- Sorting navigation is currently not working.
- No installation script currently
- Social networking integration
- No navigation hierarchy for drop-down navs.

Known bugs: #Updated 12/11/2011

- Several stylistic issues in unsupported browsers (e.g. Internet Explorer 8)
- Installation on some servers is currently very finnicky. After finishing all above functionality a lot of time will be used to ensure the system can be installed on most systems easily.
- Several 404 Errors, where functionality has not been implemented (e.g. Documentation and Social Networking)
- Some times non-alphanumeric characters get demolished by database validation.

Test Plan

An excellent resource has been providing demonstrating usage of an unit-testing plan. The development life-cycle of openjournal is a rapid schema using a modern technique called test-driven development (TDD). TDD, in short, is method of writing large-scale software with a top-down approach. With web development in specific, this typically means that you create "wishful thinking" features (without writing the code for them) and just assume that they work, and then as you realize that they do not work, you progressively enhance these features until they pass all tests.

For this area, I will focus not on the development method used, but on how we will test the end-product so that end-user will have a comprehensive and stable experience. Below is a list of functionality, mostly from the SRS above, which will be tested.

1. Dynamically store & retrieve content
2. Allow editing of content
3. Arrange stored content into categories, for editing and navigational purposes
4. Manage site navigation
5. Manage media *
6. Allow user to choose from predesigned themes
7. Manage users
8. Manage user priveleges *
9. Site maintenance mode *

* - Indicates that this feature has not yet been implemented

To test the fitness of above software specifications, we will create test cases. A CMS manages websites, so there is no better test for the system than to manage different websites with different purposes. Below are the preliminary cases that will be used:

thelittleblackbirdie.	**TechnoCloud #Updated 11/6**	**Berea College**
• a small creative business • Will be tested by Julie Davis, who is not comfortable with web design. • Will consist of few pages that aren't updated very often, and a portfolio area which is updated often. • Will have a relatively simple, but creative theme.	• a technology journalism blog • Will be tested by Sean Davis, who is relatively comfortable with web design. • Will consist of many blog-style posts, which may be updated periodically. • Theme will be simplistic, but must extend the system with new blog-oriented modules.	• a large enterprise sort of website • Will be tested by Andrew Smith who is very comfortable with web design. • Consists of many pages which aren't updated often. Categorical hierarchy is extremely important here, as is navigation management. • Will use a very intricate design.

New test cases will come up as alpha-testers come.

#Updates 11/6
Sean has agreed to be a pre-alpha tester of this system with his start-up linux blog TechnoCloud.

I am currently writing components (multitheming, blogroll, categorical navigation) which will allow each of the parties above to work with the system in a proper way. Expected completion of these components is Saturday 11/12
#Update 12/11 - These features have been created, but are untested.

Software Demo

Students in the Senior Projects course at Berea College are required to complete a video software demo, which displays the functionality of their applications. This video demo is a little over eight minutes long, and can be found at:
http://www.youtube.com/watch?v=LM5SufS5sfs

Poster session

Students are also required to attend a poster session which shows the progress made on their applications. For reference, a resized version of the poster is below!

openjournal

web content management made simple

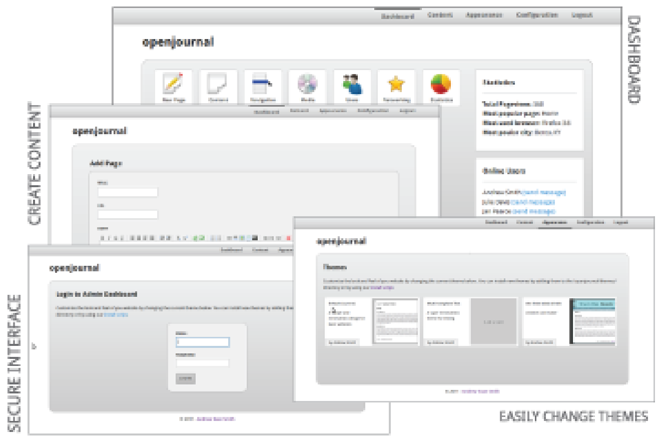
Andrew Evan Smith
CSC 493 - Senior Projects
Fall 2011

openjournal is a web application which has a simple and user-friendly focus. The system makes it easy for users to manage websites and allows them to make changes which would typically require expensive developers.

Features

1. Dynamically store and retrieve content with a secure and intuitive interface
2. Quickly and easily create or edit content
3. Manage site navigation
4. Easily install pre-designed themes!
5. Create site users for collaborative management
6. Free and open source!

Screenshots



About the Author

I'm a senior computer and information science student, as well as a freelance web designer and developer. I have been involved with several web development projects, most recently the TinyMVC web-development framework and a few large proprietary web applications using it. You can find me at any hour of the day (or night) hacking away at PHP, Python or Javascript applications both for business applications and experimentation.

My hometown is Portsmouth, Ohio, which is close enough to throw a baseball to Kentucky. I've known that I would work with new media and technology ever since I first got my hands on a computer. Every free second away from projects or course-work, I love to play racquetball, practice my archery skills, hike and spend time with my girlfriend, Julie.

Executive Section:

To: Dr. Jan Pearce, Project Director
From: Andrew Smith
Subject: openjournal.
Date: 9/7/2011

Accomplishments:
I gathered many preliminary ideas and concepts for the project. I managed to narrow down the focus of the project into a user-friendly CMS, which should fit well into the time-constraints of this course. Also, the logo & identity of the application was created.

Challenges:
It was a very difficult process narrowing the focus of the project, I would like to do it all! Fortunately, the idea has been stripped down. Also, a large challenge about starting a new project is figuring out what you're going to call it.

Time Spent:
2 hours on openjournal identity, 2 hours on project proposal, 10 minutes on the executive section.
Week total: 4 hours, 10 minutes

Goals:
Narrow the focus of the project even more, and get much needed input from colleagues about the application.

To: Dr. Jan Pearce, Project Director
From: Andrew Smith
Subject: openjournal.
Date: 9/12/2011

Accomplishments:
The vision, scope and preliminary software requirements have been drafted. This is an integral part of the project, and I'm happy with the direction these documents are leading. I also managed to start creating concepts and ideas concerning the usability of the applicatoin.

Challenges:
Many of the ideas for the project were already drafted, but not formally, so this section was not particularly challenging.

Time Spent:
2 hours on project proposal, 10 minutes on the executive section.
Week total: 2 hours, 10 minutes
To date: 6 hours, 20 minutes

Goals:
Get comments on SRS section, as well as create models for typical CMS workflow.

Continued on next page.

To: Dr. Jan Pearce, Project Director
From: Andrew Smith
Subject: openjournal.
Date: 9/19/2011

Accomplishments:
In lieu of a new task to complete today, I have focused on drafting a few concept designs for the functionality of the content-management system. These concepts are being developed into what will become the interface of the system. I have also started working on the back-end code and database for the project. Currently, I have a working draft of the database, as well as some temporary code in place for the basic CRUD operations of pages and categories. Most of this code will be changing in the future, but it at least gets me prepared to go through and develop the entire system in a coherent way. You asked us to update our Software Requirements, but I am very content with my currently, and have heard no negative comments on it.

Challenges:
A lot of my time this week has been spent further developing my ideas on the how the interface will function and on rapid prototyping of the application. The biggest challenges I've had this week involve usability and design. I will be working diligently over the coming weeks to figure out how a user can benefit most.

Time Spent:
1 hour on revisions and executive section; 10 hours on design and rapid prototyping.
Week total: 11 hours
To date: 17 hours, 20 minutes

Goals:
Further develop portions of my rapidly developed prototype and start to design the user experience of my application.

To: Dr. Jan Pearce, Project Director
From: Andrew Smith
Subject: openjournal.
Date: 9/26/2011

Accomplishments:
I started working on the design phase of this project a while ago, knowing very well that without a good, comprehensive plan, a large project like this would be impossible to manage. This week, I have taken all of my conceptual ideas, sketches and concepts and translated them into more formal digital designs. These designs include database E-R diagrams, flowcharts, directory structures, and a simple object-relationship model. All of these pieces are important, but the most important of these pieces is the openjournal functionality flowchart. This breaks down the project into chunks of modular functionality that can be managed much more easily. Also, I have a working draft of much of the content-management functionality created. This has taken a long time, and I will be testing the code to see if it is worthy of inclusion in the final project.

Challenges:
Design is a bear. As I mentioned above, design and planning can make or break your project, so I had to work very hard to come up with comprehensive designs. I struggled most with the openjournal functionality flowchart, because it encompasses so much of what a large-scale system like this should accomplish.

Time Spent:
8 hours on design and planning, 2 on programming, 30 minutes on executive section.
Week total: 10 hours, 30 minutes
To date: 27 hours, 50 minutes

Goals:
Large-scale testing of content-management portions of the system, hopeful inclusion into the final project. It is very important to design, evaluate and then redesign, so I will be reviewing my current system design to see if there are any better ideas.

To: Dr. Jan Pearce, Project Director
From: Andrew Smith
Subject: openjournal.
Date: 10/3/2011

Accomplishments:
This week I have focused completely on implementation, user-interface design and documentation. I have submitted a ZIP archive containing the system in it's current state. Please install it to see how far it has come! The biggest accomplishments I have this week is that the system is currently managing a small test website, and it's back-end interface is starting to shape up. The system manages to fully manage the content of this small test website without problems.

Challenges:
I have no specific challenges this week. Although I do have a general topic that I would like to mention. When you write the code for a system as large as this, it becomes very difficult to keep your focus when developing each part simultaneously. In normal systems this size there would be multiple developers. To accomodate this, I have focused specifically on the content-management and UI design elements of this system. Next week, I'll focus on the site configuration section.

Time Spent:
1 hour on report and executive section; 6 hours on implementation.
Week total: 7 hours
To date: 24 hours, 20 minutes

Goals:
Work on the site configuration section of the system. Also, I wish to start working on making ways that the user can be more flexible within themed layouts.

To: Dr. Jan Pearce, Project Director
From: Andrew Smith
Subject: openjournal.
Date: 10/16/2011

Accomplishments:
I have accomplished a massive amount in a short amount of time for this project. Rather than focusing on every change I made system-wide, I will give a relatively broad overview. I've applied many stylistic changes across the adminstrative dashboard, all of which are aimed to improve user-friendliness. There have been several back-end changes to the site, the biggest being the inclusion of the requirement of an account for access. Currently the system is secured by only be accessible by adminstrators with login credentials. An important note to add is that these login credentials (mostly the password) has been one-way encrypted (hashed) using md5 encryption to improve security. Another update which has not made it into the current release because it's lack of stability is different themes for different situations (e.g. a two-column theme and a one-column theme).

Challenges:
I've struggled a lot for the past few days working on is multi-file themes, in which the user can select a different type of page from his or her current theme to use for a specific page. The challenge was really conceptual, and figuring out how to design this to be both scalable and user-friendly. I'm getting close to accomplishing this.

Time Spent:
12 hours on programming, 1 hour on executive section.
Week total: 13 hours
To date: 39 hours, 50 minutes

Goals:
Next week I plan to have completed multi-page themes, as well as implementing a module which will make it easy for admins to modify how their users navigate their content.

To: Dr. Jan Pearce, Project Director
From: Andrew Smith
Subject: openjournal.
Date: 10/23/2011

Accomplishments:
Most of the accomplishments this week are non-visual. The back-end has been optimized in an extreme way, and now I am able to "plug-n-play" new dashboard sections (e.g. social networking), with little-to-no front-end work. All forms/tables are automatically generated. Although most changes were in code, there are several dashboard interface changes to ensure better usability. In this release, I've added breadcrumb navigation as well as section navigation, to ensure that the user knows how to get around.

Challenges:
The biggest challenge I've had this week is figuring out a good way for users to create and modify new navigational areas. I've got the front-end side working, but I haven't got the database end working correctly.

Time Spent:
6 hours on programming, 1 hour on executive section.
Week total: 7 hours
To date: 46 hours, 50 minutes

Goals:
Add all missing dashboard areas (managing users, etc), and work on additional functionality for users (permissions)

To: Dr. Jan Pearce, Project Director
From: Andrew Smith
Subject: openjournal.
Date: 10/30/2011

Accomplishments:
All important modules are now completed, records can now be added/deleted/updated from a modular form menu. This is very useful for quick edits, and I hope that the use of AJAX will provide the end-user with a much more pleasant experience. I've also mostly completed the navigation menu. New items can be added to a navigation, the only functionality that is currently missing is arranging the items. I hope to be able to add categories to this list as well.

Challenges:
I am starting to feel the strain of working on this large of a project alone. It rapidly becomes apparent why projects like Wordpress need dozens to hundreds of active contributors to help. The openjournal system is currently really well-structured and relatively comprehensive. The difficult comes when you try to add completely new functionality to this well-structured system, because in a way, it causes you to have to add new constraints upon how flexible the system is. A specific challenge that I've struggled with this week is with server handling and installation. Currently it's quite difficult to install openjournal on systems with varying PHP/apache configurations. I'm working very hard to let the installation be flexible relative to the server it's installed on.

Time Spent:
8 hours on programming, 1 hour on executive section.
Week total: 9 hours
To date: 55 hours, 50 minutes

Goals:
Wrap up the navigation module, and start working on more intricate details like multiple themes and statistics.

To: Dr. Jan Pearce, Project Director
From: Andrew Smith
Subject: openjournal.
Date: 11/5/2011

Accomplishments:
Unfortunately, the system is currently not workable. I am midway through implementation of a new method of multi-theming which is separate from themes as I noticed several issues with my original design. This problem is primarily that if different page types are associated with the theme, rather than the only the content, this means that if you change themes you could lose content. To fix this, I'm having a separate module called "Page Types" which will allow the user to create custom posts which can have multiple sections (e.g. Left Col, Right Col). All the database interaction and forms have been created for this new design, the only part that remains to be implemented is the relation to themes and displaying the multiple sections to a visitor of the site.

Challenges:
It was extremely difficult to come up with an intelligent way for the system to handle multiple types of pages. In sticking to a modular paradigm, I have fully separatated page types from themes, and I am working to implement it so that the end user can choose a page type, which will have different fields depending on what they've chosen. This was the hardest thing I've had to do so far on this project.

Time Spent:
10 hours on programming, 1 hour\ on executive section.
Week total: 11 hours
To date: 57 hours, 50 minutes

Goals:
Finish up multiple page types for good. Work on stabilizing the system, and then work on categorical navigation.

To: Dr. Jan Pearce, Project Director
From: Andrew Smith
Subject: openjournal.
Date: 12/11/2011

Accomplishments:
As a creator of software, it is very important to realize that your creations are never truly "finished". However, in the short timeframe of this semester, I'm very pleased with the amount of progress that I've made. I've created a functional content-management system aimed for ease of use for both designers and non-technical users. Overall the system is still in a pre-alpha state, simply because not all planned features have been developed. With that being said, all features pertaining to design and content have been created as originally planned. Users can create and edit content, as well as make decisions about how the content will be presented to the user. Designers can easily and quickly create themes for the application, with little-to-no knowledge of server-side programming.

Challenges:
I hate leaving projects unfinished. It's a very tough thing for me to do. The biggest challege for me over the past few weeks has been coming to the realization that it is literally impossible for one developer to design a feature-rich CMS to completion in a little over three months. I've learned to be satisfied with the progress that I've attained, but I will definitely not stop here. I plan on completely finishing the project and rolling out a full-release by Summer 2012.

Time Spent:
20 hours on programming, 1 hour on executive section.
Week total: 21 hours
To date: 78 hours, 50 minutes

Goals: Not to stop making progress on this system, until it is fully completed!