GospelPreaching.com

Design / Final Report

Table of Contents

Requirements Model Update....3
Overview....3
Event Table....4
Domain Model Class Diagram....5
Software Design Approach....6
Description....6
Package Diagram....7
Ask Question Sequence Diagram....7
View Article Sequence Diagram....8
Database Design....9
Overview....9
Entity Relationship Diagram....10
System Controls Design....11
Input Integrity Controls....11
Database Integrity Controls....11
Security Controls....11
Error Trapping....11
Test Cases....12
Summary of Incomplete Components....23
Installation / Access....23
Legal Notes....24
Other Notes....24

Requirements Model Update

Overview

Since the last report, the UserProfile class was dropped in favor of the built-in Django User model. It has all the fields that were required by the system and was already part of the standard authentication framework.

A “player” field was added to the Video class because the limitations of Adobe flash don't allow for a wide variety of video formats. Therefore, the “player” field was added to turn the Flash based video player on and off depending on the format of the video.

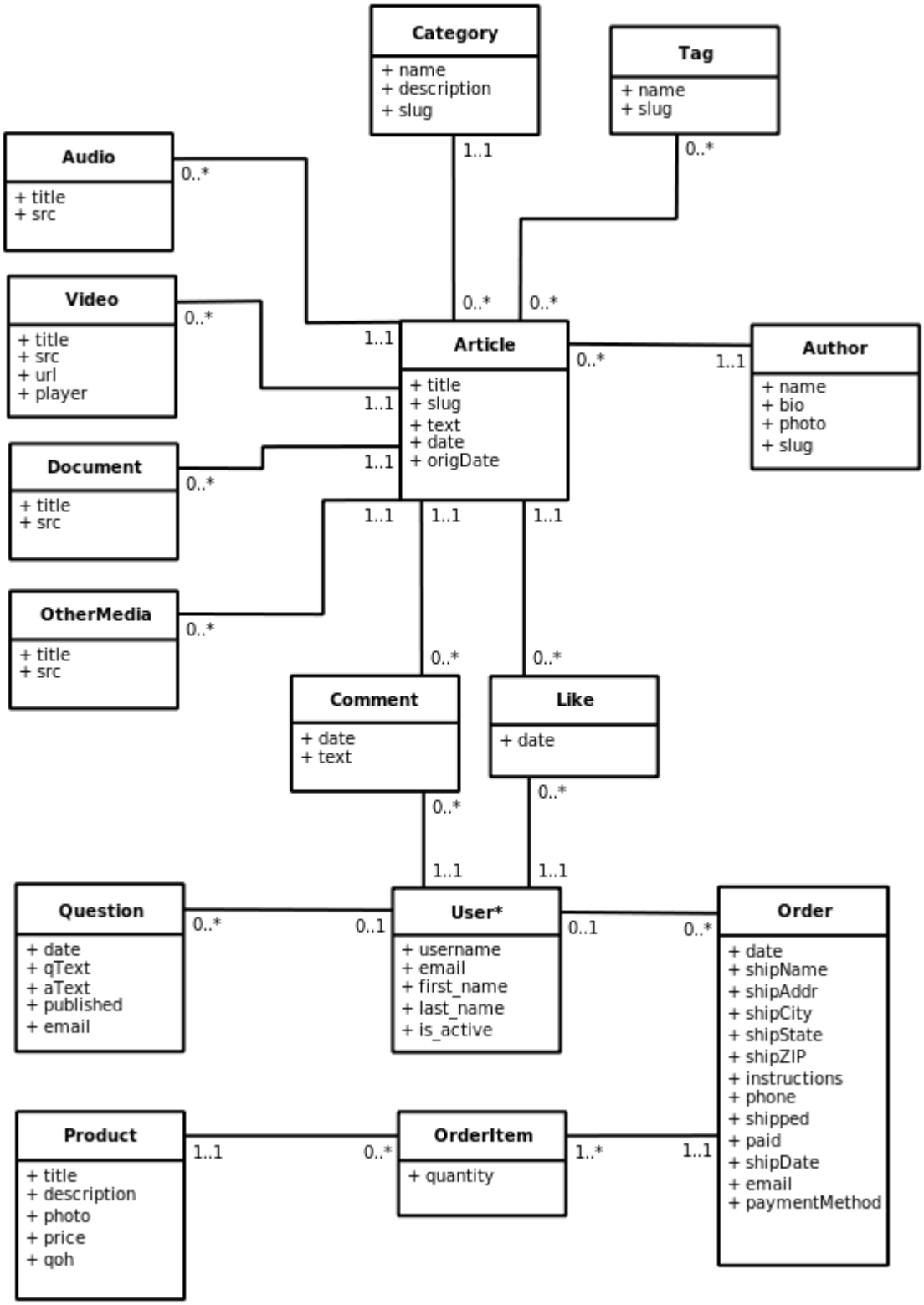
Finally, the “total” and “shippingCost” fields were dropped from the Order class since these are calculated fields that would create data redundancy if left in place. They have been replaced by methods that perform the required calculations as needed.

Create tag, update tag, delete tag were added to the Event Table. These events were overlooked on the last report.

Event Table

Event	Trigger	Source	Use Case	Response	Destination
Administrator adds an article	New article	Administrator	Add Article		
Visitor views an article	Article request	Visitor	View Article	Article details	Visitor
Administrator updates an article	Article update details	Administrator	Update Article		
Administrator deletes an article	Delete article request	Administrator	Delete Article		
Administrator adds an author	New author	Administrator	Add Author		
Visitor views an author	Author request	Visitor	View Author	Article report by Author	Visitor
Administrator updates an author	Author update details	Administrator	Update Author		
Administrator deletes an author	Delete author request	Administrator	Delete Author		
Administrator creates a category	New category	Administrator	Create Category		
Visitor views a category	Category request	Visitor	View Category	Article report by Category	Visitor
Administrator updates a category	Category update details	Administrator	Update Category		
Administrator deletes a category	Delete category request	Administrator	Delete Category		
Administrator creates a tag	New Tag	Administrator	Create Tag		
Visitor views tagged articles	Article request by tag	Visitor	View Tagged Articles	Report of tagged articles	Visitor
Administrator updates a tag	Tag update details	Administrator	Update Tag		
Administrator deletes a tag	Delete tag request	Administrator	Delete Tag		
Visitor wants to register a new user account	New user	Visitor	Register User	Registration confirmation New user notice	Visitor Administrator
Administrator approves a new user account	User account approval	Administrator	Approve New User	Account approval notice	Visitor
Visitor views a user profile	User profile request	Visitor	View User Profile	User profile details	Visitor
Visitor updates their profile	User profile update details	Visitor	Update User Profile		
Administrator or Visitor deletes a user account	Delete user request	Administrator or Visitor	Delete User		
Visitor adds a new comment	New comment	Visitor	Add Comment		
Administrator or Visitor removes a comment	Delete comment request	Administrator or Visitor	Remove Comment		
Visitor likes an article	Like article request	Visitor	Like An Article		
Visitor unlikes an article	Unlike article request	Visitor	Unlike An Article		
Visitor submits a question	New question	Visitor	Ask Question	New question notice	Administrator
Visitor views a question	Question request	Visitor	View Question	Question details	Visitor
Administrator updates a question	Question update details	Administrator	Update Question		
Administrator deletes a question	Delete question request	Administrator	Delete Question		
Administrator adds a new product	New product	Administrator	Add Product		
Visitor views product	Product request	Visitor	View Product	Product details	Visitor
Administrator updates a product	Product update details	Administrator	Update Product		
Administrator deletes a product	Delete product request	Administrator	Delete Product		
Visitor places an order	New order	Visitor	Create New Order	Order confirmation New order notice	Visitor Administrator
Administrator records an order as being shipped	Record order as shipped request	Administrator	Record Order As Shipped	Shipping confirmation	Visitor
Administrator adjusts an order	Order update details	Administrator	Adjust Order		
Administrator requests report of recent orders	Recent order report request	Administrator	View Recent Orders	Recent order report	Administrator

Domain Model Class Diagram



** User class is part of the Django authentication framework.*

Software Design Approach

Description

The application architecture I am using is a modified version of the MVC approach to web development unique to the Django platform.

The models, or data access takes place in models.py. No SQL strings are ever directly manipulated by user created code; the Django framework handles all of that. The structure of the classes is simply defined in the model and all the SQL string conversion are taken care of automatically based on the connection settings in the main settings.py file.

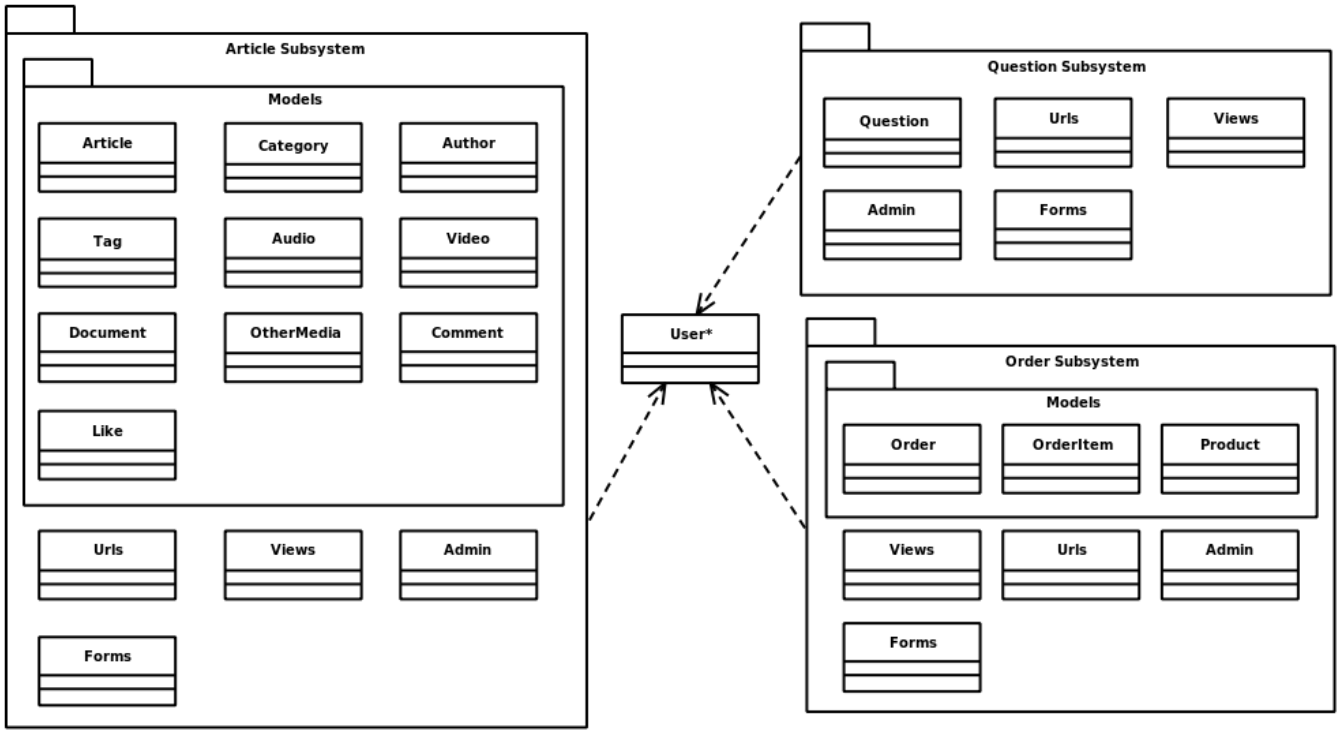
The controller code is mostly handled by the framework. However, there are configuration settings that manage the functionality of the controller layer. The most notable of these is the urls.py file which defines which view to display.

According to the Django's interpretation of MVC, the views.py file manages the view layer. This interpretation defines the view layer as making the decision of ***what*** to display, instead of ***how*** to display it. This logic would traditionally be found in the controller layer in other MVC frameworks, which is one area where Django differs.

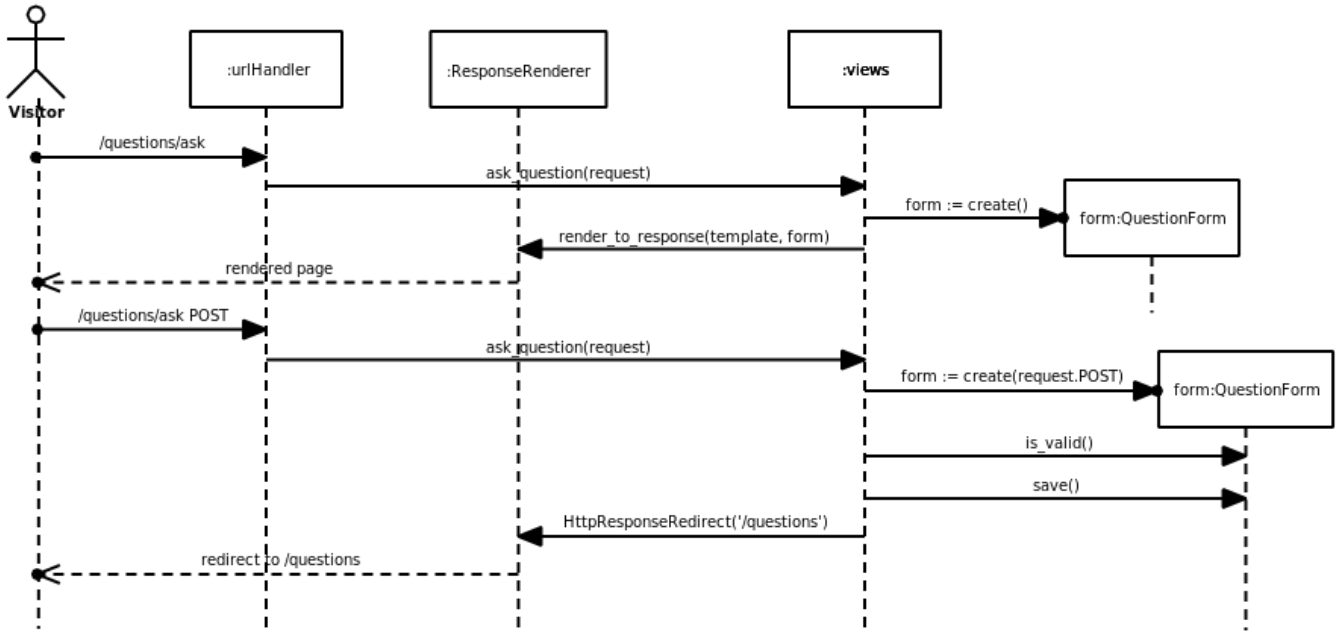
The final piece of the puzzle is the template system. The templates define *how* the information is to be displayed, and consist of HTML files with template tags. However, Django does not consider these to be part of any of the three layers.

The code for the system will strongly object-oriented and dynamic thanks to the Python language.

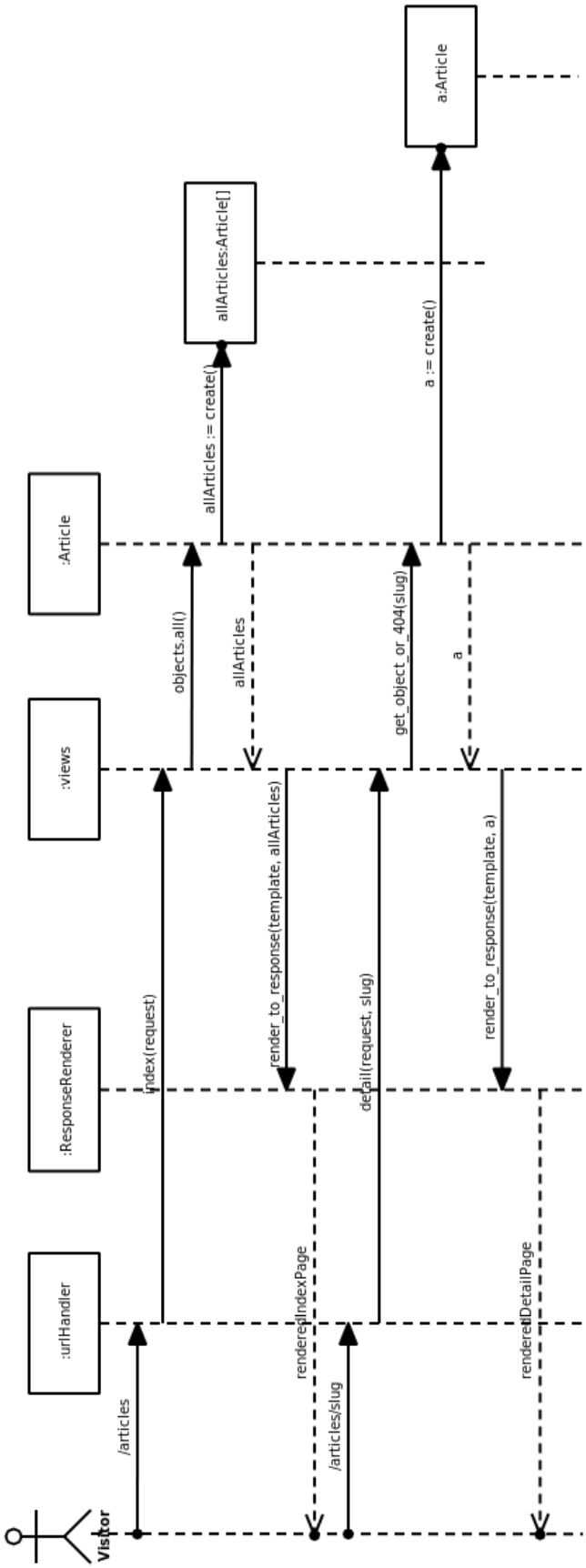
Package Diagram



Ask Question Sequence Diagram



View Article Sequence Diagram

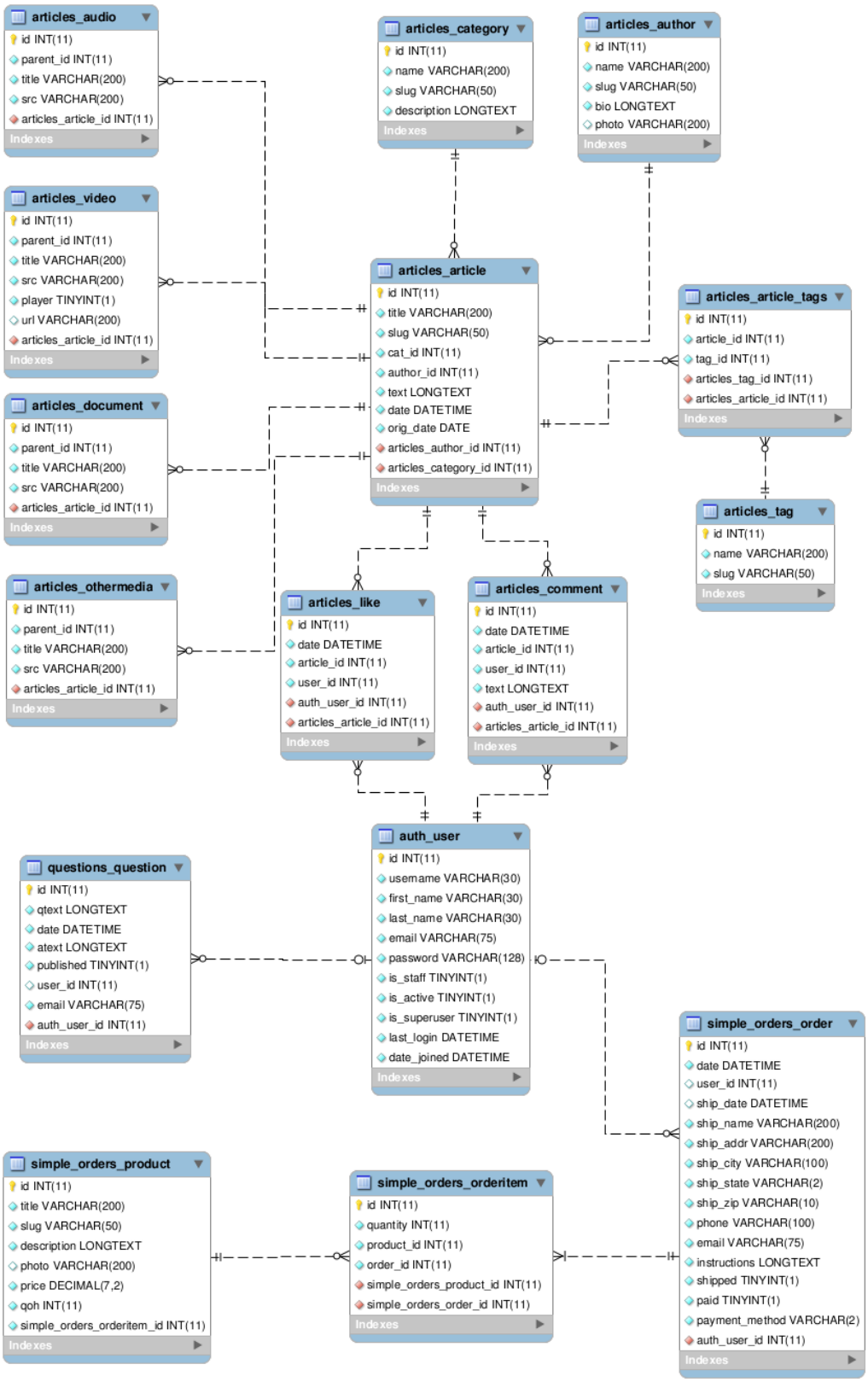


Database Design

Overview

The system will use a MySQL database that resides on the same web server that the system will run on. The Django framework will handle all table creation and management tasks.

Entity Relationship Diagram



System Controls Design

Input Integrity Controls

The system's input comes in the form of web forms submitted through POST. These forms are defined as classes in the Django framework. Each form I have defined is created as a class with several field objects. Each of these field objects has requirements set up that define what valid input for that field is. The form itself will sometimes have definitions of requirements for the form as a whole. When the POST data is joined back with the form object, it places each value in the appropriate field and then the view will call is_valid() on the form object to check to see if the inputs have passed validation. If the form indicates that all the requirements have been met, the view will continue processing the information, otherwise it will return the form to the user with notes on what values need to be adjusted.

Database Integrity Controls

The system uses the Django framework's model system to handle database integrity. The system's model code contains information on referential integrity and the project's settings file contains connection information including a username and password for the database. Transactions are handled automatically within the model system, but can be overridden if need be in special situations by the view code.

Database backup and recovery will be handled through the web host's administration panel. This will involve having the web master download regular backups of the database and uploaded files for restoration in cases of data loss.

Security Controls

The system implements Django's authentication and session systems for security. The authentication system allows for simple user management and permissions. If a view needs to check to make a user is authenticated before continuing to process information for a request, I simply check request.user.is_authenticated(). I can also use the @login_required decorator to force the user to authenticate before granting access to that entire view. User information is stored in cookies with the help of the session system. This system actually stores the information on the server side and places an encrypted cookie on the user's machine for unlocking the session data on the server side. I use this system for the user authentication as well as for storing shopping cart information.

Error Trapping

Error trapping is handled through the use of server error pages and the project's debug mode. If the debug mode is on, the server will display a page containing complete debugging information so that a developer can diagnose problems. If the debug mode is off, it will display standard server error pages, customized to the current graphical design. An example of this would be when the user attempts to visit a page that does not exist and gets a 404 error page.

Test Cases

- Add Article
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the article management section of the article application admin area. Administrator clicks “Add article”. Administrator enters the article details as follows:
  - Validation should prevent a blank entry for all of the following fields: title, slug, category, author, date originally created.
  - Validation should prevent a slug that already exists in the database.
  - Validation should prevent a slug containing anything other than alphanumeric characters, dashes, and underscores; for example: “test article” should cause a validation error, prompting the user to edit that field.
  - Validation should prevent the user from entering anything other than a proper date in the date originally created field; for example: “apple” should cause a validation error, prompting the user to edit that field.
  - The administrator should enter the following information to obtain a proper, validated article: title “Test Article”, slug “test-article”, category “Test Category” (which should exist in the system already and be pre-populated in the drop-down), author “David Nichols” (which should exist in the system already and be pre-populated in the drop-down), text “This is a test article.”, date originally created “2010-05-14”, tags “Test Tag” (which should exist in the system already and be pre-populated in the selection area).
  - The article should now appear on the site and in the articles_article table in the database.
- View Article
  - The user visits the site and clicks on the “Articles” link on the menu. User selects an article from the list provided. The article's details and related media items should display in the browser.
  - If the user enters a URL for an article that doesn't exist, like “http://dev.gospelpreaching.com/articles/this-article-doesnt-exist”, then the server should return a 404 error page.

- Update Article
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the article management section of the article application admin area. Administrator clicks on the “Test Article” that he created earlier. Administrator updates the article by adding an audio file in the “Audio” section of the form with the title “Test Audio”.
  - The Update form should have the same validation rules as the Add Article form above.
  - The update should be reflected on the site and in the articles_article table in the database.
- Delete Article
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the article management section of the article application admin area. Administrator checks the box next to the “Test Article” that he created earlier. Then he selects “Delete selected articles” from the Actions drop down at the top of the page and clicks “Go”. The article should now be removed from the list and the articles_article table in the database.
- Add Author
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the author management section of the article application admin area. Administrator clicks “Add author”. Administrator enters the author details as follows:
  - Validation should prevent a blank entry for the following fields: name, slug.
  - Validation should prevent a slug that already exists in the database.
  - Validation should prevent a slug containing anything other than alphanumeric characters, dashes, and underscores; for example: “test author” should cause a validation error, prompting the user to edit that field.
  - Validation should prevent the user from adding anything other than a proper image file to the photo field.
  - The administrator should enter the following information to obtain a proper, validated author: name “Test Author”, slug “test-author”, bio “This is a test author.”.
  - The author should now appear on the site and in the articles_author table in the database.

- View Author
  - The user visits the site and clicks on the “Articles” link on the menu. User clicks on the “Author” button at the top of the list and types “test” into the search box. The “Test Author” should appear in the list below the search box. The user clicks on the “Test Author” and is directed to the author detail page where the author's name, bio, photo, and related articles (if available) are displayed.
  - If the user enters a URL for an author that doesn't exist, like “http://dev.gospelpreaching.com/articles/author/this-author-doesnt-exist”, then the server should return a 404 error page.
- Update Author
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the author management section of the article application admin area. Administrator clicks on the “Test Author” that he created earlier. Administrator updates the author by changing the bio to “This bio has been changed.”
  - The Update form should have the same validation rules as the Add Author form above.
  - The update should be reflected on the site and in the articles_author table in the database.
- Delete Author
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the author management section of the article application admin area. Administrator checks the box next to the “Test Author” that he created earlier. Then he selects “Delete selected authors” from the Actions drop down at the top of the page and clicks “Go”. The author should now be removed from the list and the articles_author table in the database.

- Create Category
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the category management section of the article application admin area. Administrator clicks “Add category”. Administrator enters the category details as follows:
  - Validation should prevent a blank entry for the following fields: name, slug.
  - Validation should prevent a slug that already exists in the database.
  - Validation should prevent a slug containing anything other than alphanumeric characters, dashes, and underscores; for example: “test category” should cause a validation error, prompting the user to edit that field.
  - The administrator should enter the following information to obtain a proper, validated category: name “Test Category”, slug “test-category”, description “This is a test category.”.
  - The category should now appear on the site and in the articles_category table in the database.
- View Category
  - The user visits the site and clicks on the “Articles” link on the menu. User clicks on the “Category” button at the top of the list selects the “Test Category” from the list. The user is directed to the category detail page where the category's name, description, and related articles (if available) are displayed.
  - If the user enters a URL for a category that doesn't exist, like “http://dev.gospelpreaching.com/articles/category/this-category-doesnt-exist”, then the server should return a 404 error page.
- Update Category
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the category management section of the article application admin area. Administrator clicks on the “Test Category” that he created earlier. Administrator updates the category by changing the description to “This description has been changed.”
  - The Update form should have the same validation rules as the Add Category form above.
  - The update should be reflected on the site and in the articles_category table in the database.

- Delete Category
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the category management section of the article application admin area. Administrator checks the box next to the “Test Category” that he created earlier. Then he selects “Delete selected categories” from the Actions drop down at the top of the page and clicks “Go”. The category should now be removed from the list and the articles_category table in the database.
- Create Tag
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the tag management section of the article application admin area. Administrator clicks “Add tag”. Administrator enters the tag details as follows:
  - Validation should prevent a blank entry for the following fields: name, slug.
  - Validation should prevent a slug that already exists in the database.
  - Validation should prevent a slug containing anything other than alphanumeric characters, dashes, and underscores; for example: “test tag” should cause a validation error, prompting the user to edit that field.
  - The administrator should enter the following information to obtain a proper, validated tag: name “Test Tag”, slug “test-tag”.
  - The tag should now appear on the site and in the articles_tag table in the database.
- View Tagged Articles
  - The user visits the site and clicks on the “Articles” link on the menu. User clicks on the “Tag” button at the top of the list and types “test” into the search box. The “Test Tag” should appear in the list below the search box. The user clicks on the “Test Tag” and is directed to the tag detail page where the tag's name and related articles (if available) are displayed.
  - If the user enters a URL for a tag that doesn't exist, like “http://dev.gospelpreaching.com/articles/tag/this-tag-doesnt-exist”, then the server should return a 404 error page.
- Update Tag
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the tag management section of the article application admin area. Administrator clicks on the “Test Tag” that he created earlier. Administrator updates the tag by changing the name to “Test Tag Edited”, and the slug to “test-tag-edited”.
  - The Update form should have the same validation rules as the Add Tag form above.
  - The update should be reflected on the site and in the articles_tag table in the database.

- Delete Tag
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the tag management section of the article application admin area. Administrator checks the box next to the “Test Tag” that he created earlier. Then he selects “Delete selected tags” from the Actions drop down at the top of the page and clicks “Go”. The tag should now be removed from the list and the articles_tag table in the database.
- Register User
  - A user who wants to register for an account on the system will navigate to the login page, and click on the “User Registration” link. This will bring up the user registration form. The user will fill out the form as follows:
  - All fields are required. Any blank fields should return the user to the form and indicate that the field is required.
  - Validation will check to make sure that the username meets the following requirements: 30 characters or fewer, Alphanumeric characters only (letters, digits and underscores). A username of “Boomer Bear” will cause this validation error.
  - Validation will ensure that both password fields match. Otherwise the user will be returned to the form with a validation error.
  - Validation will ensure that the email field is a valid email address. For example, “boomer.missouristate.edu” will return the user to the form with a validation error because of the lack of an “@” sign.
  - The user should enter the following information to create a proper, validated user registration: username “bbear”, password & password confirmation “Mascot1”, email “boomer@missouristate.edu”, first name “Boomer”, last name “Bear”.
  - The user should now appear in the user administration area and in the auth_users table in the database. The new user should be listed with the is_active field set to false.
- Approve New User
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the user management section of the auth application admin area. Administrator filters the user list by “active”, “no”. Administrator clicks on the user he wants to approve (“bbear”), which brings up the user update page. The administrator checks the user's active box and clicks save.
  - The user should now be able to log in and will appear in the user list on the site.

- View User Profile
  - The user visits the site and clicks on the “Users” link on the menu. This brings up a list of active user accounts on the system. The user clicks on the “Boomer Bear” account and is directed to the user profile page where the information is displayed.
  - If the user enters a URL for a user that doesn't exist or is inactive, like “http://dev.gospelpreaching.com/accounts/profile/notauser”, then the server should return a 404 error page.
- Update User Profile
  - The user visits the site and logs in. The user then clicks on the “Profile” link on the menu. This takes the user to their profile page where they are presented with a form to update their account info. The user (“bbear”) changes his email to “boomer@live.missouristate.edu”.
  - The Update form should have the same validation rules as the User Registration form above.
  - The update should be reflected on the site and in the auth_user table in the database.
- Delete User
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the user management section of the auth application admin area. Administrator checks the box next to the “bbear” user that was created earlier. Then he selects “Delete selected users” from the Actions drop down at the top of the page and clicks “Go”. The user should now be removed from the list and the auth_user table in the database.
- Add Comment
  - The user (“bbear”) logs in to the site and navigates to an article (“Test Article”). The user scrolls down to the comment form beneath the article and types in his comment. After hitting submit, the user will be sent back to the “Test Article” page where the comment now appears with his name “Boomer Bear” and the date.
  - The comment should also appear on the user's profile page and in the articles_comment table in the database.
- Remove Comment
  - The user logs in the the site and goes to either his profile page or the article he commented on. Once he locates his comment he can click the “delete” link next to it to remove the comment.
  - The user should only be able to delete his own comments.
  - An administrator should be able to delete any comment.
  - Once deleted, the comment will not appear on the site, and will no longer be in the articles_comment table in the database.

- Like an Article
  - The user (“bbear”) logs in to the site and navigates to an article (“Test Article”). The user scrolls down to the “like” section beneath the article and clicks the “like this article” link. The user will be sent back to the “Test Article” page where the like now appears with his name “Boomer Bear”.
  - The like should also appear on the user's profile page and in the articles_like table in the database.
- Unlike an Article
  - The user logs in the the site and goes to the article he liked. He can click the “unlike this article” link in the “like” section to remove the like.
  - The user can only delete his own likes.
  - Once deleted, the comment will not appear on the site, and will no longer be in the articles_like table in the database.
- Ask Question
  - A user who wants to ask a question will click on the “Q & A” link on the menu, and then on the “Ask a question” link. This will bring up the question form. The user will fill out the form as follows:
  - The “question text” field is required. A blank “question text” field should return the user to the form and indicate that the field is required.
  - If the user is logged in, there should only be a “question text” field. If not, there will also be an optional “email” field
  - Validation will ensure that the email field is a valid email address. For example, “boomer.missouristate.edu” will return the user to the form with a validation error because of the lack of an “@” sign.
  - The logged in user (“bbear”) should enter the following information to create a proper, validated question: question text “I have a test question.”.
  - The question should now appear in the question administration area and in the questions_question table in the database. The new question should be listed with the published field set to false.
- View Question
  - The user visits the site and clicks on the “Q & A” link on the menu. This brings up a list of published questions on the system. The user clicks on the “I have a test question.” question and the question is expanded to display the response.

- Update Question
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the question management section of the questions application admin area. Administrator clicks on the “I have a test question.” that was created earlier. Administrator updates the question by adding the answer text “This is a test answer.”, and checking the “published” box.
  - The Update form should have the same validation rules as the Ask a Question form above.
  - The update should be reflected in the questions_question table in the database and on the site, with the question and answer now appearing in the “Q & A” section.
- Delete Question
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the question management section of the questions application admin area. Administrator checks the box next to the “I have a test question.” that was created earlier. Then he selects “Delete selected questions” from the Actions drop down at the top of the page and clicks “Go”. The question should now be removed from the list and the questions_question table in the database.
- Add Product
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the product management section of the simple_orders application admin area. Administrator clicks “Add product”. Administrator enters the product details as follows:
  - Validation should prevent a blank entry for the following fields: title, slug, price, quantity on hand.
  - Validation should prevent a slug that already exists in the database.
  - Validation should prevent a slug containing anything other than alphanumeric characters, dashes, and underscores; for example: “test product” should cause a validation error, prompting the user to edit that field.
  - Validation should prevent the user from adding anything other than a proper image file to the photo field.
  - Validation should require the price and quantity on hand fields to be valid numbers. Therefore, a value of “apple” in either field should return a validation error.
  - The administrator should enter the following information to obtain a proper, validated product: title “Test Product”, slug “test-product”, description “This is a test product.”, price “10.00”, quantity on hand “10”.
  - The product should now appear on the site and in the simple_orders_product table in the database.

- View Product
  - The user visits the site and clicks on the “Store” link on the menu. User selects an product from the list provided. The product's details should display in the browser.
  - If the user enters a URL for an product that doesn't exist, like “http://dev.gospelpreaching.com/store/products/this-product-doesnt-exist”, then the server should return a 404 error page.
- Update Product
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the product management section of the simple_orders application admin area. Administrator clicks on the “Test Product” that he created earlier. Administrator updates the product by changing the quantity on hand to “20”
  - The Update form should have the same validation rules as the Add Product form above.
  - The update should be reflected on the site and in the simple_orders_product table in the database.
- Delete Product
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the product management section of the simple_orders application admin area. Administrator checks the box next to the “Test Product” that was created earlier. Then he selects “Delete selected products” from the Actions drop down at the top of the page and clicks “Go”. The product should now be removed from the list and the simple_orders_product table in the database.

- Create New Order
  - A user who wants to place an order will click on the “Store” link on the menu, and click on the product they wish to purchase (“Test Product”). This will bring up the product detail page. The user will how many of the product they wish to purchase (“3”), and repeat for all the products they want to place in this order. Once they are ready to place the order, the user will click on the “Cart” link on the menu. Here they can adjust the products and quantities they wish to purchase. Once they are happy with the selection, the user will click the “Checkout” link and be redirected to the checkout page with a final order form.
  - All fields are required except for the additional instructions field. Any blank entries in these fields should return the user to the form and indicate that the field is required.
  - Validation will ensure that the email field is a valid email address. For example, “boomer.missouristate.edu” will return the user to the form with a validation error because of the lack of an “@” sign.
  - The user should enter the following information to create a proper, validated user registration: shipping name “Boomer Bear”, shipping address “901 S. National Ave.”, shipping city “Springfield”, shipping state “MO”, shipping zip “65897”, phone “417-836-5000”, email “boomer@missouristate.edu”, payment method “Check”.
  - The order should now appear in the order administration area and in the simple_orders_order table in the database.
- Record Order as Shipped
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the order management section of the simple_orders application admin area. Administrator filters the order list by “shipped”, “no”. Administrator checks the box next to the order that was created earlier. Then he selects “Mark selected orders as shipped” from the Actions drop down at the top of the page and clicks “Go”. The product should now be marked as shipped and be updated in the simple_orders_order table in the database.
- Adjust Order
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the order management section of the simple_orders application admin area. Administrator clicks on the order that he created earlier. Administrator updates the order by changing the checking the paid box.
  - The Update form should have the same validation rules as the Create New Order form above.
  - The update should be reflected in the admin area and in the simple_orders_order table in the database.

- View Recent Orders
  - Administrator logs on to the system and clicks on the “Admin” link on the menu. Administrator navigates to the order management section of the simple_orders application admin area. Administrator uses the date navigation links at the top of the list to narrow the list down to a certain period, such as the current month.
- User Login
  - The user clicks on the “Login” link in the menu and is redirected to the login form.
  - Username and password are required. If the username and password do not match an entry in the auth_user table (password is hashed for database storage) then the user will be returned to the page and informed that the login was unsuccessful.
  - To successfully login with the Boomer Bear account created earlier, the user would enter the following in the form: user name “bbear”, password “Mascot1”.
- User Logout
  - The logged in user clicks on the “Logout” link in the menu.
- User Changes Password
  - The user visits the site and logs in. The user then clicks on the “Profile” link on the menu. This takes the user to their profile page where they click on the “Change Password” link. This take the user to the password change form.
  - The old password field must match the previous password, and the new password and new password confirmation fields should match.
  - To successfully change the password of the Boomer Bear account used earlier, the user would enter the following in the form: old password “Mascot1”, new password “HungyBear2”, new password confirmation “HungyBear2”.
  - The user should be able to login with the new password, and the hash should be updated in the database.

Summary of Incomplete Components

The only component that wasn't finished was the PayPal integration, due to recent API changes.

Installation / Access

The system's code can be accessed at http://github.com/mybluevan/gospel-preaching/tree/v0.1.

The system is set up on a public development server for testing. You can access it by going to http://dev.gospelpreaching.com. I have set up a user with administration rights for you to test the system. The username is “satzinger” and the password is “cis591”.

Legal Notes

Other Notes

I want to give credit to the amazing Django web framework and thank the amazing people that work on it. “It lets you build high-performing, elegant Web applications quickly.” Please check out the Django project's site: <http://www.djangoproject.com/>