GospelPreaching.com

HOLY BIBLE

System Background

- CFTF Publications is a small publisher of religious books, tracts, and other media related to the Church of Christ.

- Allen Bailey is the owner/operator.

System Background

- The system is going to be an online portal for:

  - CFTF's electronic resources (documents, audio, video, etc.) and

  - a storefront for the company's published goods.

- GospelPreaching.com
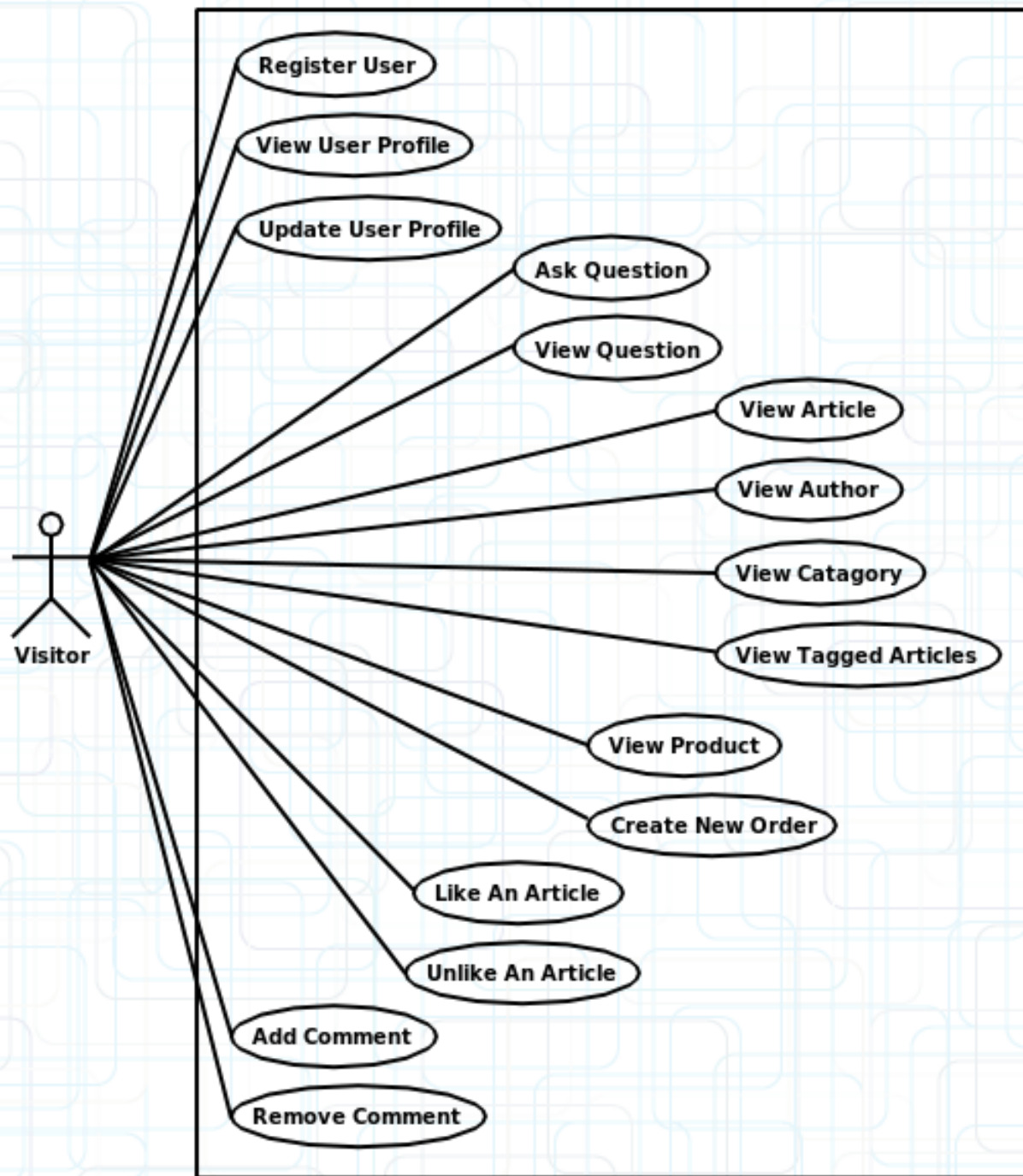
System Background

- Along with the system's basic functions it will also provide the following:

  - Ability to categorize and tag electronic materials

  - Post and respond to questions (Q&A)

  - Allow users to connect their profiles with social networking sites (Facebook, Twitter, etc.)

  - Post comments
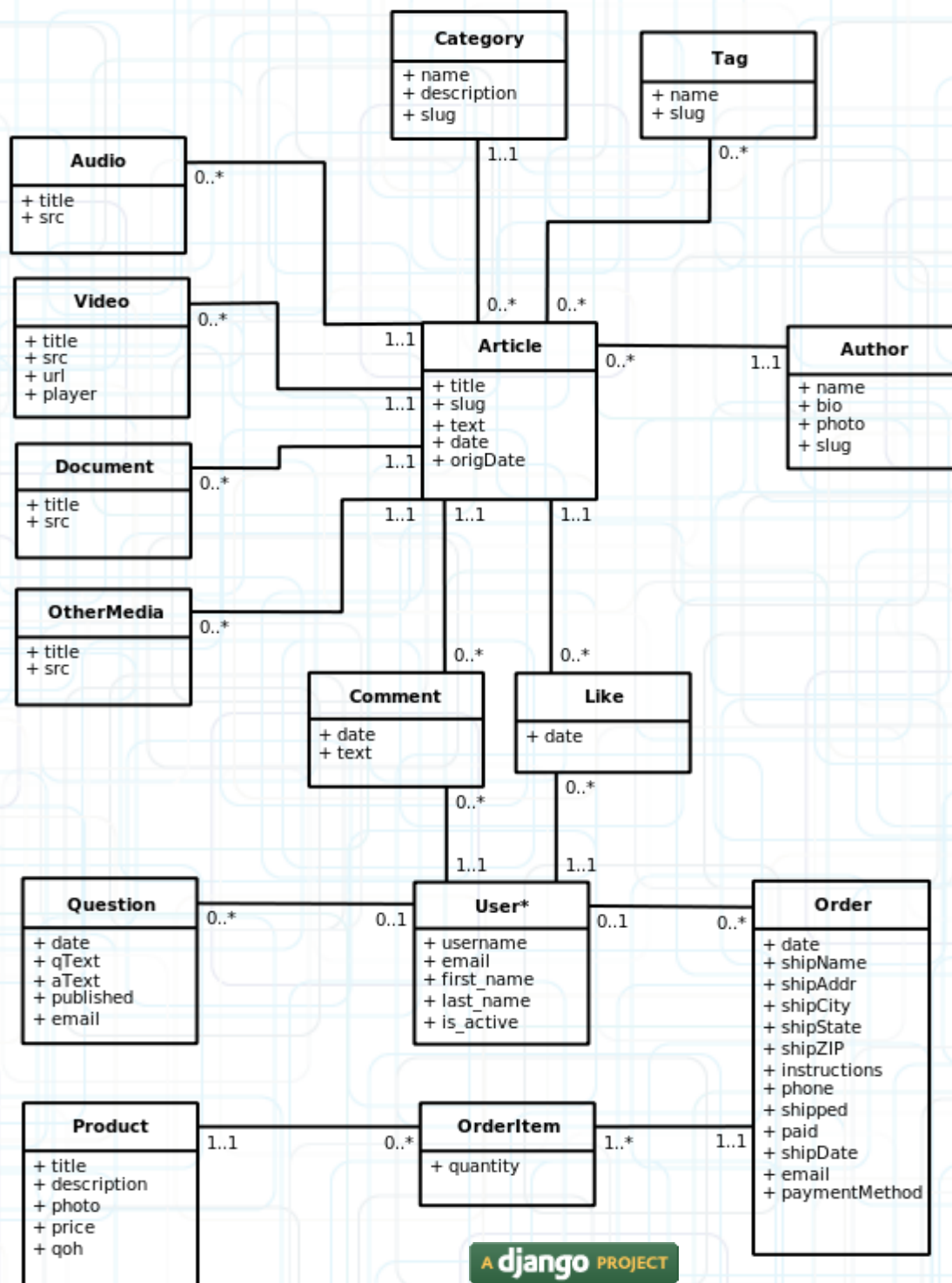
  - "Like" articles.

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		
Visitor views tagged articles	Article request by tag	Visitor	View Tagged Articles	Report of tagged articles	Visitor
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

Use Case Diagrams

A django PROJECT

Domain Model Class Diagram

Category
+ name
+ description
+ slug

Tag
+ name
+ slug

Audio
+ title
+ src

Video
+ title
+ src
+ url
+ player

Document
+ title
+ src

OtherMedia
+ title
+ src

Article
+ title
+ slug
+ text
+ date
+ origDate

Author
+ name
+ bio
+ photo
+ slug

Comment
+ date
+ text

Like
+ date

Question
+ date
+ qText
+ aText
+ published
+ email

User*
+ username
+ email
+ first_name
+ last_name
+ is_active

Order
+ date
+ shipName
+ shipAddr
+ shipCity
+ shipState
+ shipZIP
+ instructions
+ phone
+ shipped
+ paid
+ shipDate
+ email
+ paymentMethod

Product
+ title
+ description
+ photo
+ price
+ qoh

OrderItem
+ quantity

Multiplicities: Audio 0..*; Video 0..*; Document 0..*; OtherMedia 0..*; Category 1..1 / 0..*; Tag 0..*; Article 1..1, 1..1, 1..1, 1..1, 1..1, 0..*, 1..1; Author 1..1; Comment 0..*; Like 0..*; User* 1..1; Question 0..*; Order 0..1, 0..*; Product 1..1; OrderItem 0..*, 1..*, 1..1

Technology and Working Environment

- 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.

Technology and Working Environment

- 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.

Technology and Working Environment

- 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.

- I've provided a sequence diagram for the View Article use case to illustrate this structure as well as possible.

Technology and Working Environment

- Development Machine

- SVN Server

- Publicly Accessible Dev Servers

  – Local Django test server

  – Public FastCGI server

System Controls

- Error trapping

- Form Validation

- Referential integrity checks

Demonstration

A django PROJECT

Demonstration

- Anonymous User
  - Viewing of models

- User with a User Account
  - Adding/Removing comments
  - Liking/Unliking articles

Demonstration

- Administrator
  - Article Administration
  - Question Administration
  - User Administration
  - Product Administration
  - Order Administration