

System Design Document

Article Database Application

By A2 Team (Anjana and Alfred)

Index

1. Architecture

1.1 Web Framework

2. Data Access Design

2.1 Database Design (Model Design)

2.2 Functional Decomposition (Main CRUD Operations)

2.3 Database Investigation

3. Security Design

3.1 Framework Security

3.1.1 Framework Security Features

3.1.2 Security Features used in Article Database Application Project

3.2 Security Mechanisms

3.3 User Roles and Permissions

4. User Interface Design

4.1 View Design (User interfaces required for each user stories)

4.1.1 ... As a student...

4.1.2 ...As one of the tutors...

4.1.3 ...As the administrator (Principle)...

4.2 UI Design

4.2.1 UI and UX practices in Design:

4.2.2 Portability

4.2.3 Public Interface Wire Frame and Admin Interface Markup

4.2.3.1 Public Interface Wire Frame

4.2.3.2 Admin Interface Markup

1. Architecture

1.1 Web Framework

The solution will be built using a Django *Content Management System (CMS)* which provides the engine for the website, and allows a simple and fast way to develop a first working solution.

The diagram below gives an overview of one CMS architecture, for the Django framework, which is based on the Model-View –Controller pattern.

The CMS provides developers with templates for:

- The models (which are the entities to be stored in the database)
- The views (which represent the user interface or web pages that get displayed on the browser)

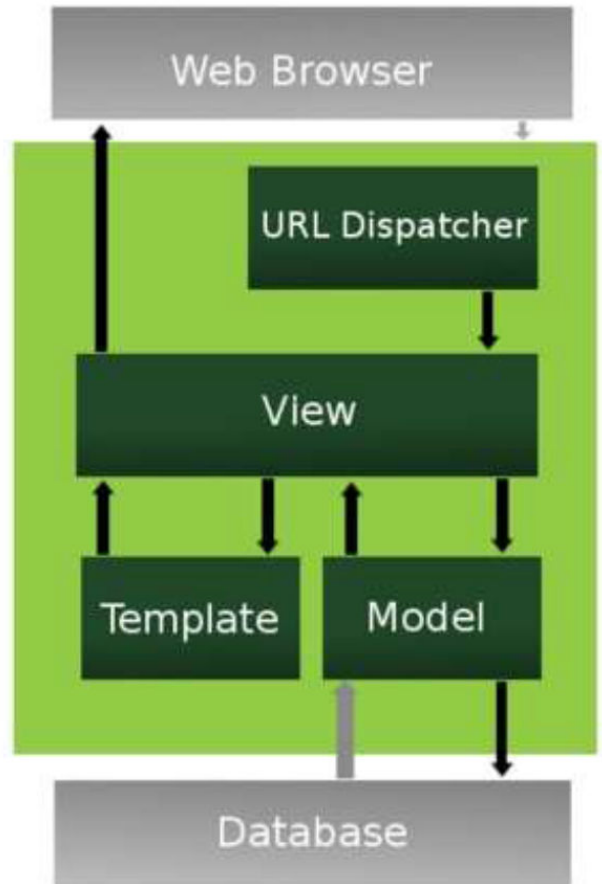
The CMS provides

- The controller through its framework (light green box) and the URL dispatcher

Developers will define the Views (based on templates) and the Models and the CMS will provide the rest.

Image Source: fishwarter (2009). The Django Web Application Framework. Retrieved from <https://www.slideshare.net/fishwarter/the-django-web-application-framework-2-1221388>

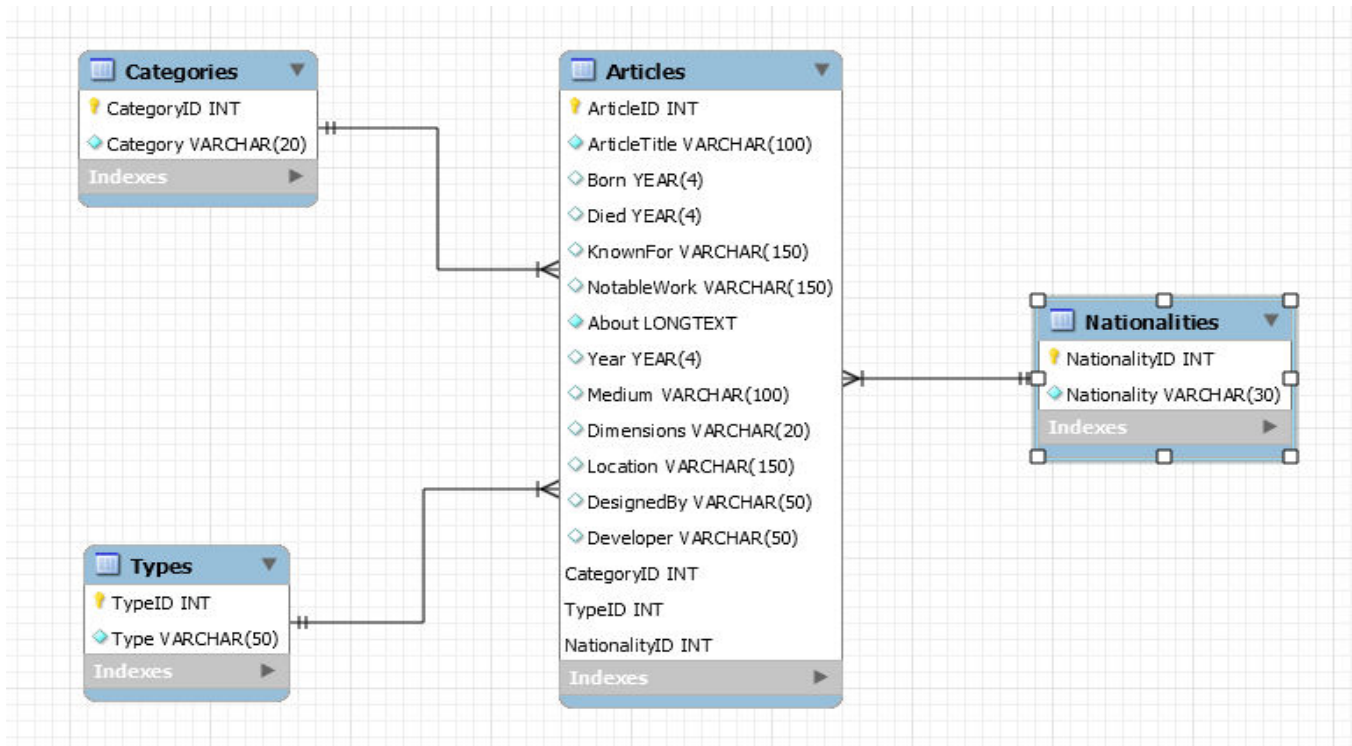
The sections below will document the Model and Views that are required for this application.



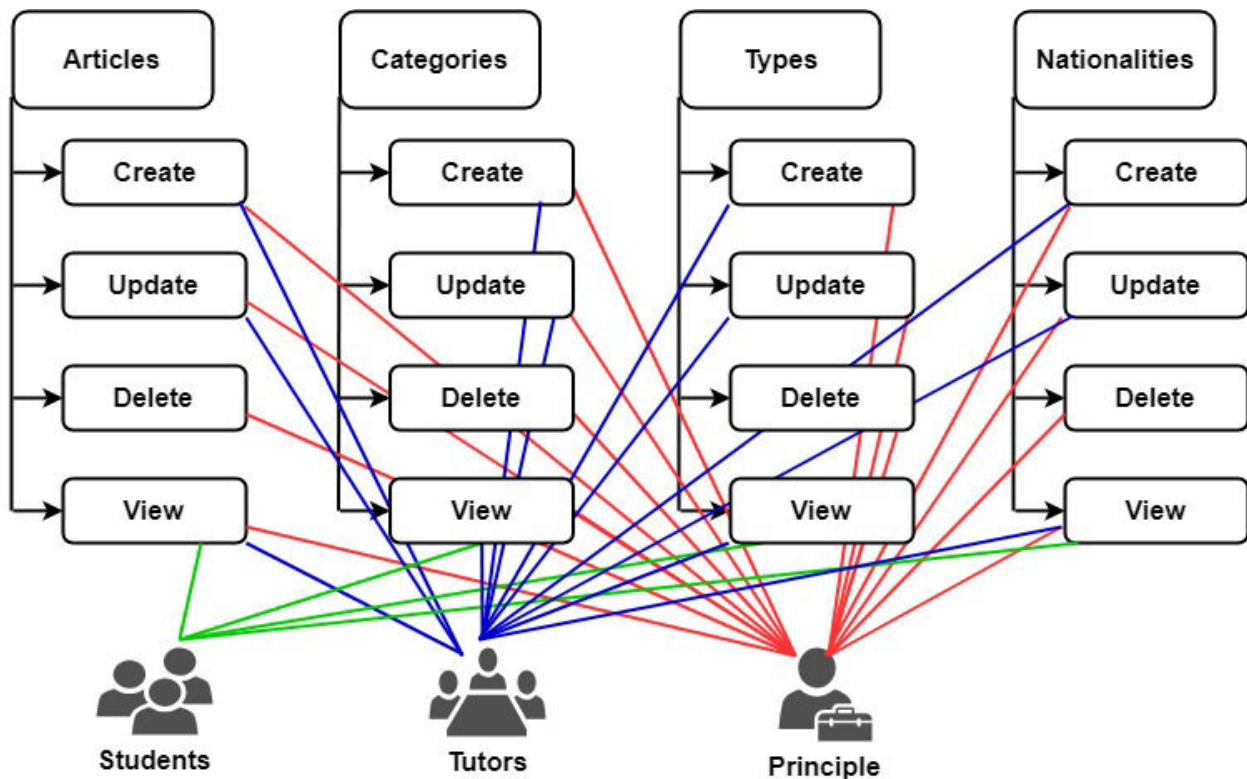
2. Data Access Design

2.1 Database Design (Model Design)

Database model as at the end of the second iteration of the Article Database Application.



2.2 Functional Decomposition (Main CRUD Operations)



2.3 Database Investigation

We have considered MySQL database management system and MongoDB database management system for this web app. We have decided to use MySQL for this Article Database Application project.

Reasons for using MySQL are justified in below table:

Evaluation Criteria	Django based Custom-built web application with MySQL Database	Django based Custom-built web application with MongoDB
Cost	Medium	Low
Timeliness	Fast	Late
Support Resources and Documentation	Many	Less
Database Scalability	Medium	High
Risk Level	Low security risk	Medium security risk
Required Technical Skills	Medium - needs to be outsourced	High - needs to be outsourced
Industry Recognition	Well known technology	Immerging technology
Overall Ranking	1	2

3. Security Design

3.1 Framework Security

3.1.1 Framework Security Features

- Cross site scripting (XSS) protection
- Cross site request forgery (CSRF) protection
- SQL injection protection
- Clickjacking protection
- SSL/HTTPS
- Host header validation
- Session security
- Form data validation and filtering
- Password protection and encryption

3.1.2 Security Features used in Article Database Application Project

- Cross site scripting (XSS) protection
- Cross site request forgery (CSRF) protection
- SQL injection protection
- Session security
- Password protection and encryption

3.2 Security Mechanisms

- Cross site scripting (XSS) protection - Using Django templates protects our app against the majority of XSS attacks. However it is possible to turn off this protection.
- Cross site request forgery (CSRF) protection - The way the protection is enabled is that we include the {% csrf_token %} template tag in our form definition. This token is then rendered in our HTML with a value that is specific to the user on the current browser.
- SQL injection protection - In almost every case we'll be accessing the database using Django's querysets/models, so the resulting SQL will be properly escaped by the underlying database driver.
- Session security mechanisms - can be established by using Django inbuilt session modules they can be activated by placing following code in settings.py file.

```
INSTALLED_APPS = [  
    'django.contrib.sessions',]  
  
MIDDLEWARE = [  
    'django.contrib.sessions.middleware.SessionMiddleware',]
```

- Password protection and encryption - can be established by using Django inbuilt authentication modules they can be activated by placing following code in settings.py file.

```
INSTALLED_APPS = [  
    'django.contrib.auth',  
    'django.contrib.contenttypes',]  
  
MIDDLEWARE = [  
    'django.contrib.sessions.middleware.SessionMiddleware',  
    'django.contrib.auth.middleware.AuthenticationMiddleware',]
```

3.3 User Roles and Permissions

- Students – are allowed to search and view article records online.
- Tutors – are allowed to add, edit and view article records online.
- Administrator (Principle) - is allowed to add, edit, view and delete article records online.

4. User Interface Design

4.1 View Design (User interfaces required for each user stories)

4.1.1 ... As a student I want to search and view article records online, so I can do my studies and researches quickly and efficiently.

Pages:

- Home Page
- Search Result Page
- Article Details Page

4.1.2 ...As one of the tutors I want to add, edit and view article records online, so as to meet regular students' needs.

Pages:

- Admin Log-in Page (Admin Site)
- Admin Log-out Page (Admin Site)
- Manage Articles Page (Delete option excluded) (Admin Site)
- Manage Categories Page (Delete option excluded) (Admin Site)
- Manage Types Page (Delete option excluded) (Admin Site)
- Manage Nationalities Page (Delete option excluded) (Admin Site)

4.1.3 ...As the administrator (Principle) I want to add, edit, view and delete article records online, so I can keep the web application in consistent order in daily basis.

Pages:

- Admin Log-in Page (Admin Site)
- Admin Log-out Page (Admin Site)
- Manage Articles Page (Delete option included) (Admin Site)
- Manage Categories Page (Delete option included) (Admin Site)
- Manage Types Page (Delete option included) (Admin Site)
- Manage Nationalities Page (Delete option included) (Admin Site)

4.2 UI Design

4.2.1 UI and UX practices in Design:

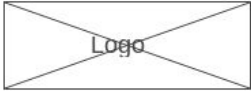
- Easy to use User Interface by simplicity of content placement and layout design.
- Well organized site navigation should be implemented.
- Site Colors: Navigation - #8D5CE8, Header & Footer - Black
- Font-family: 'Arimo', sans-serif;

4.2.2 Portability

- The design must be responsive and multiple devices compatible.
For students' number one priority - mobile phone interface.
For staff member's number one priority – desktop and tablet.

4.2.3 Public Interface Wire Frame and Admin Interface Markup

4.2.3.1 Public Interface Wire Frame

	School Name
<div>Search</div> <div>Category Drop Down</div> <div>Category Drop Down</div> <div>Category Drop Down</div>	Article Search Result
Footer	

4.2.3.2 Admin Interface Markup


Django administration

WELCOME, UBUNTU. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)


Site administration

AUTHENTICATION AND AUTHORIZATION

Groups


+ Add  Change

Users


+ Add  Change

CATALOG


Articles

+ Add  Change


Types

+ Add  Change

Categories

+ Add  Change

Nationality

+ Add  Change

Recent actions

My actions

None available