Project Report On

Library Management System

Branch- Computer Science and Engineering(B.tech)

Submitted By -

Name	Roll Number
ONKAR CHUGH	**17075039**
AMAN YADAV	**17075007**

INTRODUCTION

The Project

To create an online web application for the purpose of efficient and effective Library Management System in any institute for registered students.

Problem Statement

We know that as in IIT-(BHU), for issuing books, students have to go to the Main library (which has around 1.5 lakh books) and then find the books they wish to borrow. Although having separate compartments for similar kind of books based on department and genres, the process of browsing through the large shelfs in the physical library could be cumbersome.

In order to resolve all these problems, we decided to create this project.

Salient Features

- **User-friendly:** Our website can be used by both students (borrower) and the librarian. We have separate portals for both.

- **Easy surfing:** Books can be searched based on author, book genre, details, etc.

- **E-library concept:** Students can issue their books online without being physically present and the librarian can manage book records and student accounts online.

REQUIREMENT SPECIFICATION

1. Django version 2.1.3
2. Python 3.7.0
3. django-crispy-forms
4. MySQL

RELATIONAL SCHEMA

In our project, we have implemented tables using models. We have the following tables:

1. GENRE: This defines the book category. Attributes are:

- name- just to name the category.

2. LANGUAGE: The language in which the book is published. Attributes are:

- name- the language name.

3. BOOK: This relation contains information about a book. Attributes are:

- title- title of the book.
- Author- writer of the book content. Also a foreign key from “Author”.
- Summary- brief description about the book.
- Isbn- unique identification code for a particular copy of a book.

- Genre- category.
- Language- It is the foreign key from “language” in models.
- total_copies- number of copies of a particular type of book which is owned by the library.
- available_copies- number of copies of a particular type of book which are remaining un-issued.

4. AUTHOR: Relation containing details of the author of the book. Attributes are:

- first_name
- last_name
- date_of_birth
- date_of_death

5. STUDENT: Relation containing details of students. Attributes are:

- roll_no- this is chosen as the primary key.
- Name
- branch
- contact_no
- total_books_due

6. BORROWER: Relation containing borrowing details. Attributes are:

- student- foreign key from “student”.

- Book- foreign key from “book”.
- issue_date
- return_date- date of book return.

PROJECT STRUCTURE

- Our main project folder is “Library_management”.
- We have an app named “management”.
- Book records are located in management_book
- auhor details located in management_author
- forms.py contain model form-BookForm, BorrowForm, StudentForm, RatingForm

FUNCTIONALITIES

We have used a single app in our project named ‘management’.It contains all the database,template and views.

Functions in views.py :

- **index** – renders the home page index.html
- **login_view**- renders the login page
- **logout_view**-redirects user to index.html when user log out
- **register_view**- as the name suggest it renders the registration page ‘register.html’

- **BookListView**- list all the books available in library and renders 'book_list.html'
- **BookDetailView**-accepts a parameter pk ie the id of book renders the detail of book to 'book_detail.html'
- **Student_BookListView**-render the books issued by the user.
- **BookCreate**-renders for for adding a new book.
- **BookUpdate**- accepts a parameter pk ie the id of book and render update form for book.
- **BookDelete**-accepts a parameter pk i.e the id of book superuser want to delete.
- **Student_request_issue**-issues a book to student if he hasn't issued more than 10 books.
- **StudentCreate**-renders a form for user creation
- **StudentUpdate**-accepts a parameter pk i.e the id of student renders a form for updating information of book

- **StudentDelete**- accepts a parameter pk i.e the id of student and delete the student record from database
- **StudentDetail**- accepts a parameter pk i.e the id of student and renders the information of student
- **ret**- accepts a parameter pk i.e the id of borrower object and delete the entry from database.
- **RatingUpdate**- accepts a parameter pk i.e the id of Reviews object and render a form for updating it
- **RatingDelete**- accepts a parameter pk i.e the id of Reviews object and delete the entry from database.

USE OF MIDDLEWARE

We have used a middleware “GoogleSearch”. Whenever any syntatical error in the program codes is witnessed during the execution of website, a new tab automatically opens with a google search of that error and giving the user a crystal-clear idea about the error and how can the user fix it.

USE OF SYNDICATION

Syndication in the form of RSS feeds is used to update students about new trending, interesting and useful books.

REFERENCES

We used the following resources to our convenience. We would like to thank their authors whole-heartedly:

1. Python docs.
2. Django documentation.
3. Tango with Django 1.9/1.10/1.110

Conclusion

Our project is available on Github.

It is completely open source and can be seen here.

https://github.com/AMAN2202/Django-Library-Management-System

Also it is deployed online .Have a look

http://aman2202.pythonanywhere.com/

THANK YOU!