Django Library Management System

This is Django based web application named as Library Management system App.

Django Server: It is inbuilt server that is given by Django itself.it default run on port:8000.

Create Django Project- First we have installed Django.

Then create Django project using command- Django-admin start project project name.

Create Django App- jump on project folder create app using command-

python manag.py startapp appname

I have created 1 project and 2 apps:

Create a database **Django dB** in MySQL, and configure into the **settings.py** file of Django project. See the example.

// settings.py

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'djangodb',
        'USER':'root',
        'PASSWORD':'mysql',
        'HOST':'localhost',
        'PORT':'3306'
    }
}
```

Create a Model

Put the following code into **models.py** file.

Django Library Management System

// models.py

```
from django.db import models
class Employee(models.Model):
    id = models.CharField(max_length=20)
    name = models.CharField(max_length=100)
    email = models.EmailField()
    contact = models.CharField(max_length=15)
    class Meta:
        db_table = "employee"
```

Create a ModelForm

// forms.py

```
from django import forms
from employee.models import Employee
class EmployeeForm(forms.ModelForm):
    class Meta:
        model = Employee
        fields = "__all__"
```

Create View Functions

// views.py

// urls.py

Organize Templates

// Base.html

// update.html

Django Library Management System

Now we have a 2 Apps and some Description about all:

1: LibraryManagementSystem:

files:

setting.py:

1. add apps in INSTALLED_APP
2. give MySQL database connectivity
3. give a template base directory path

1: library_management_app:

files:

1 - models.py – It is basically Model. That contains class

class name: BookModel

class variable field:

title: that contain title of a book.

Description: that contain description of book

2 - views.py – It is basically View we logic for data CRUD. That contains functions.

Function name: base

Argument: request object

Return: render on home page

Function name: book_list

Argument: request object

Return: render on booklist page and show model data

Function name: add_book

Argument: request object

Return: render on create book page and show model data form

for add book

Function name: update_book

Argument: request object, id(int)

return: render on update_book page and show modeldata form for update_book.

Function name: delete_book

Argument: request object, id(int) //delete book by given id

Return: redirect to booklist page

3 - forms.py – It is basically Form. That contains class

class name: BookForm

About: it creates form for modelBook

4 - urls.py

About: In this file I have given a URL for HTML page.

1: Book Service:

files:

1 - models.py – It is basically Model. That contains class

class name: CustomerUser // create customusermodel for email login

class variable field:

email: Takes email as Field

password: Takes password as Field

2 - views.py – It is basically View we logic for data CRUD. That contains functions

Function name: signup

Argument: request object

return: render on signup page

3 - forms.py – It is basically Form. That contains class

class name: CustomerUserForm

About: it create form for model CustomUser

4 - urls.py

About: In this file I have given a URL for HTML page (Signup, Login, Student).

Server Run Command- python manage.py runserver

ADMIN-

we can create admin using command: python manage.py createsuperuser

1.admin signup: using email, password, confirm password.

2.admin login: using email and password.

Finally, My Library Management App is Completed.

GitHub repository link is given here: vivekpurohit13/Library-Management-system_app-: Crud Operation on Libreary management Using Django rest API (github.com)

"THANK YOU"