**Library Management System**

**INTRODUCTION:**

1. **Introduction:**

A School or College Library is where students can have access to a variety of resources that are all kinds of books. Students can register themselves to this system. The admin can then issue them with any book they want to read for a period of 14 days. The students are requested to return it within 14 days (2 weeks) of time or else there is a Rs 5 fine each day till the book is being returned. After returning they again issue a new book if the student wants.

1. **Aims And Objectives:**

The main objective of this project is to manage the details of the students, books and the issued books efficiently. This project helps to keep all the records of the library.

1. **System Requirements:**

3.1Front End - HTML

3.2 Back end - Python Django, MySQL

3.3 Tools Used - Visual Studio

1. **Project Features:**

#### **4.1 For admin:**

* Admin can add new books
* View the whole list of books and delete any book
* View the whole list of students and delete any student
* Issue a book to a student
* View all the issued books

#### 4.2 For students:

* A student can see his/her profile
* Edit their profile
* Can change password
* View the issued book by them

**APPENDIX :**

Following are the Different Objects used in our project -

* **from django.db import models** -
* A model is the single, definitive source of information about your data. Each model is a Python class that subclasses [django.db.models.Model](https://docs.djangoproject.com/en/4.1/ref/models/instances/" \l "django.db.models.Model" \o "django.db.models.Model).
* Each attribute of the model represents a database field.
* With all of this, Django gives you an automatically-generated database-access API .
* **from django.contrib.auth.models import User -**

This configuration has evolved to serve the most common project needs, handling a reasonably wide range of tasks, and has a careful implementation of passwords and permissions.

* **from datetime import datetime,timedelta -**

**timedelta()** function is present under*datetime library* which is generally used for calculating differences in dates and also can be used for date manipulations in Python. It is one of the easiest ways to perform date manipulations.

* **def** \_\_str\_\_(self):

It is a python method which is called when we use print/str to convert object into a string.

* ****Book model -****

It saves all the data of each book added to the system.

* ****Student model -****

It saves all the details of each student registered to the system.

* ****IssuedBook model -****

It saves all the details of any book issued by a student.

* **from django.urls import path -**

Returns an element for inclusion in url patterns.

* **from . import views -**

Django views are Python functions that takes http requests and returns http response, like HTML documents. A web page that uses Django is full of views with different tasks and missions. Views are usually put in a file called views.py located on your app's folder.

* **def index(request) -**

The request parameter is a HttpRequest object, which contains data about the request

* **return render(request, "index.html") -**

This function uses the render() function to create the HttpResponse that is sent back to the browser.

* **<form method="POST"> -**

In the post method, after the submission of the form, the form values will not be visible in the address bar of the new browser tab as it was visible in the GET method. It appends form data inside the body of the HTTP request. It has no size limitation.