LIBRARY MANAGEMENT SYSTEM

DESIGN DOCUMENT

TEJ PATEL

DATABASE DESIGN CS 6360

23-OCT-2017

Version-1

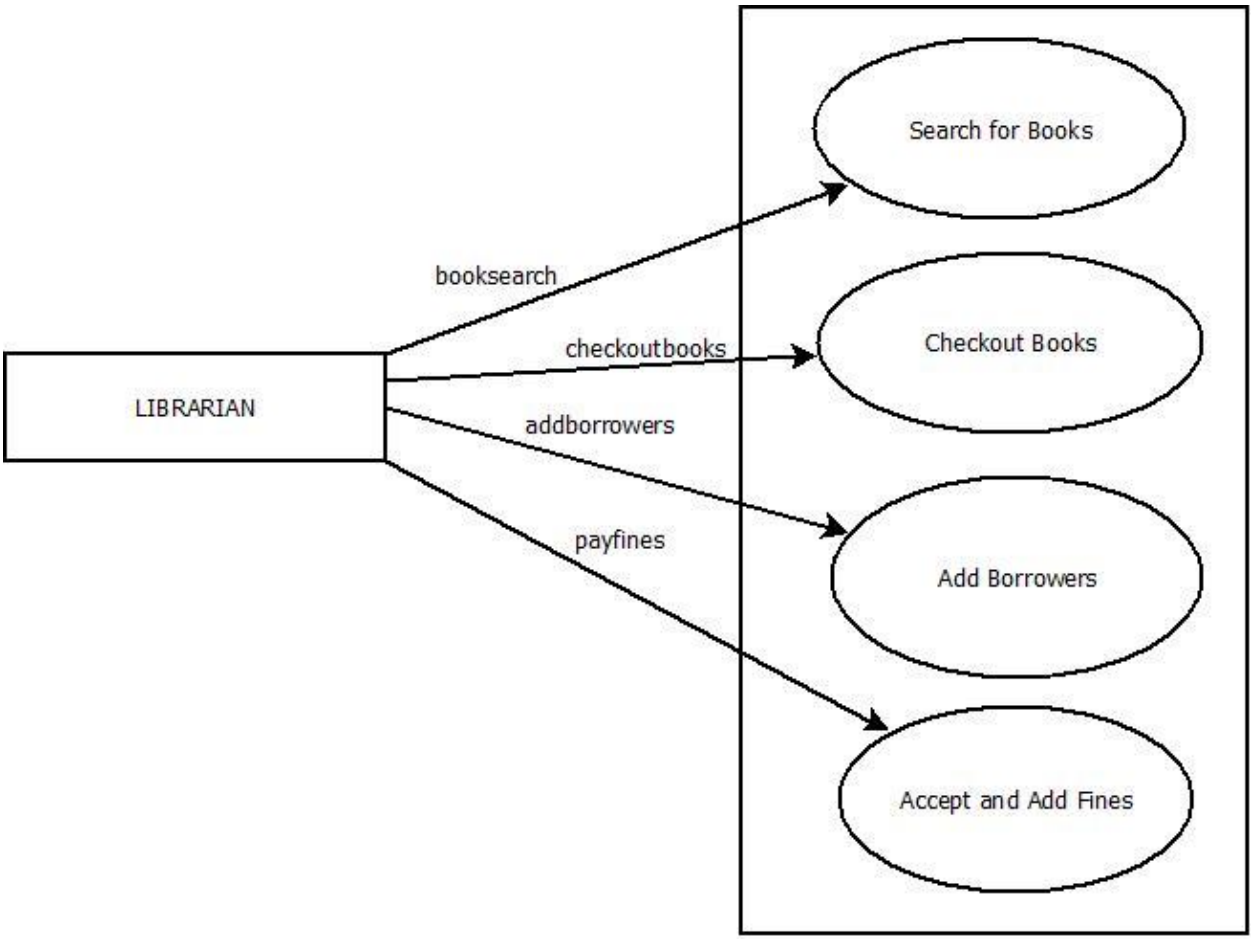
Table of Contents

Introduction

Daily a lot of people go to library to read books. Libraries are an important part of all schools and colleges and are present in all of those. Huge volumes of people go to library to gain knowledge by reading and checking out books. So to keep track of all the transactions is nearly impossible for a human as there are simply a lot of books and people to keep track of. So, this software is designed to assist the developers to manage all these in a much better way.

Actors and Use Case

The main actor is the librarian and there are many different things that a librarian can do with the software. Librarian can search for a book, checkout and check-in books, add borrowers to system and accept the fines from borrowers and feed it into the system. Here is the use case diagram to give some visualization.



Design Overview

This software is a website that can run on local web server inside a library and possibly on internet. The front-end is developed in HTML and CSS along with Bootstrap, JQuery and Javascript. The backend is developed in python using Django. MYSQL database is used as a DBMS. Djangp uses MVC framework.

Data Design and Assumptions

The schema used is same as provided in the description with some minute changes. One extra column, 'Availability', is added in 'Book' schema to keep track of availability of books. Note that this can be done without addition of this column by checking if 'Date_In' is NULL in 'Book_Loans' table but that would create some extra overhead as additional 2 queries are needed to run. ISBN13 information is not used as not given in the Schema and requirements as well.

Architecture

Django has **MVC** architecture:

The **model(M)** is a model or representation of your data. It's not the actual data, but an interface to the data. The model allows you to pull data from your database without knowing the intricacies of the underlying database. The model usually also provides an *abstraction* layer with your database, so that you can use the same model with multiple databases.

The **view(V)** is what you see. It's the presentation layer for your model. On your computer, the view is what you see in the browser for a Web app, or the UI for a desktop app. The view also provides an interface to collect user input.

The **controller(C)** controls the flow of information between the model and the view. It uses programmed logic to decide what information is pulled from the database via the model and what information is passed to the view. It also gets information from the user via the view and implements business logic: either by changing the view, or modifying data through the model, or both.

Libraries Used

Frontend : HTML, CSS, Bootstrap, JQuery, Javascript

Backend : Django, pymysql, mysqlclient, pandas

Database: MySQL