Online Library Management System with Python, Flask & MySQL

webdamn.com/online-library-management-system-with-python-flask-mysql

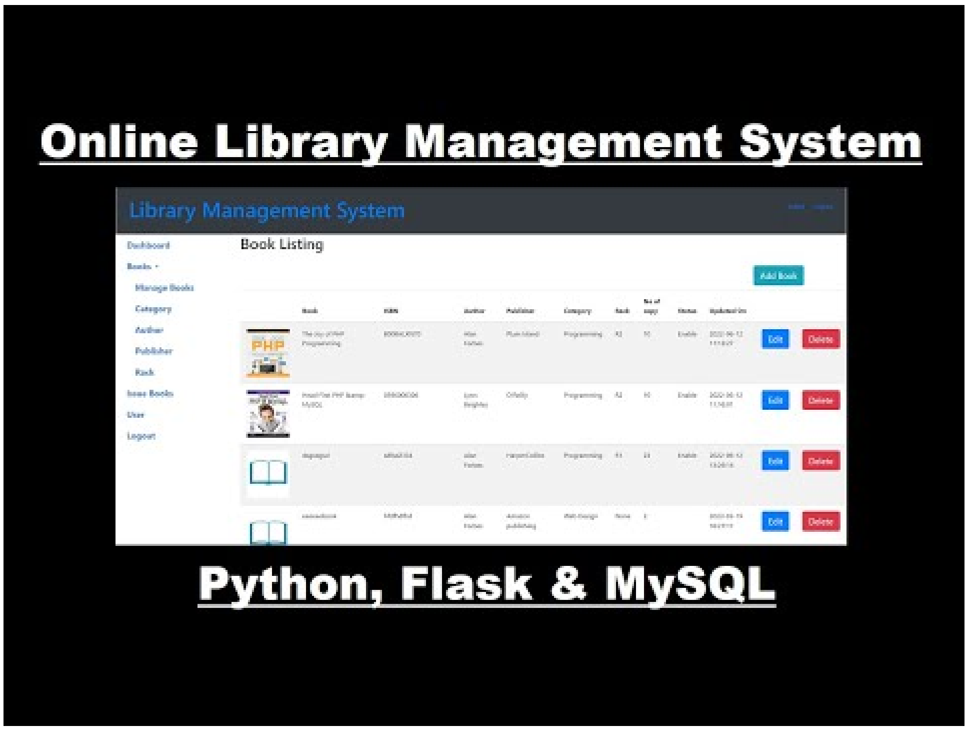
webdamn

March 26, 2023

In our previous Python tutorial, we have explained how to develop User Management System with Python, Flask and MySQL. In this tutorial, we will explain how to develop Online Library Management System with Python, Flask and MySQL.

Online Library Management System is an automated system that handles the various functions of the library such as manage books, issue books, manage users to access system etc.

So let's proceed to develop Online Library Management System with Python, Flask and MySQL.



Watch Video At: https://youtu.be/37hHjWF3a0k

Application Setup

We will create application directory `library-management-system-python` using below command.

```
$ mkdir library-management-system-python
```

we moved to the project direcotry

```
$ cd library-management-system-python
```

Install Required Module

We will use folloing modules for this application.

- **Flask**: It is a micro framework from Python to create web application. So we will install this module to create web applications. We will install it using the below command:

  ```
  pip install Flask
  ```

- **flask_mysqldb**: This is Python package that can be used to connect to MySQL database. We will install it using the below command:

  ```
  pip install flask_mysqldb
  ```

Create MySQL Database and Table

We will create MySQL database `library-system` and create table `user` to store users login details to access system.

```
CREATE TABLE `user` (
  `id` int(11) UNSIGNED NOT NULL,
  `first_name` varchar(255) DEFAULT NULL,
  `last_name` varchar(255) DEFAULT NULL,
  `email` varchar(255) DEFAULT NULL,
  `password` varchar(64) NOT NULL,
  `role` enum('admin','user') DEFAULT 'admin'
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER TABLE `user`
  ADD PRIMARY KEY (`id`);

ALTER TABLE `user`
  MODIFY `id` int(11) UNSIGNED NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=6;
```

we will create table `book` to store book details.

```
CREATE TABLE `book` (
  `bookid` int(11) NOT NULL,
  `categoryid` int(11) NOT NULL,
  `authorid` int(11) NOT NULL,
  `rackid` int(11) NOT NULL,
  `name` text COLLATE utf8_unicode_ci NOT NULL,
  `picture` varchar(250) COLLATE utf8_unicode_ci NOT NULL,
  `publisherid` int(11) NOT NULL,
  `isbn` varchar(30) COLLATE utf8_unicode_ci NOT NULL,
  `no_of_copy` int(5) NOT NULL,
  `status` enum('Enable','Disable') COLLATE utf8_unicode_ci NOT NULL,
  `added_on` datetime NOT NULL DEFAULT current_timestamp(),
  `updated_on` datetime NOT NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

ALTER TABLE `book`
  ADD PRIMARY KEY (`bookid`);

ALTER TABLE `book`
  MODIFY `bookid` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=6;
```

we will create table `issued_book` to store issue book details.

```
CREATE TABLE `issued_book` (
  `issuebookid` int(11) NOT NULL,
  `bookid` int(11) NOT NULL,
  `userid` int(11) NOT NULL,
  `issue_date_time` datetime NOT NULL DEFAULT current_timestamp(),
  `expected_return_date` datetime NOT NULL,
  `return_date_time` datetime NOT NULL,
  `status` enum('Issued','Returned','Not Return') COLLATE utf8_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

ALTER TABLE `issued_book`
  ADD PRIMARY KEY (`issuebookid`);

ALTER TABLE `issued_book`
  MODIFY `issuebookid` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;
```

we will create table `category` to store books category.

```
CREATE TABLE `category` (
  `categoryid` int(11) NOT NULL,
  `name` varchar(200) COLLATE utf8_unicode_ci NOT NULL,
  `status` enum('Enable','Disable') COLLATE utf8_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

ALTER TABLE `category`
  ADD PRIMARY KEY (`categoryid`);

ALTER TABLE `category`
  MODIFY `categoryid` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=8;
```

we will create table `author` to store book author details.

```
CREATE TABLE `author` (
  `authorid` int(11) NOT NULL,
  `name` varchar(200) COLLATE utf8_unicode_ci NOT NULL,
  `status` enum('Enable','Disable') COLLATE utf8_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

ALTER TABLE `author`
  ADD PRIMARY KEY (`authorid`);

ALTER TABLE `author`
  MODIFY `authorid` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;
```

we will create table `publisher` to store book publisher details.

```
CREATE TABLE `publisher` (
  `publisherid` int(11) NOT NULL,
  `name` varchar(255) NOT NULL,
  `status` enum('Enable','Disable') NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

ALTER TABLE `publisher`
  ADD PRIMARY KEY (`publisherid`);

ALTER TABLE `publisher`
  MODIFY `publisherid` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=11;
```

we will also create table `rack` to store book rack details.

```
CREATE TABLE `rack` (
  `rackid` int(11) NOT NULL,
  `name` varchar(200) COLLATE utf8_unicode_ci NOT NULL,
  `status` enum('Enable','Disable') COLLATE utf8_unicode_ci NOT NULL DEFAULT
'Enable'
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

ALTER TABLE `rack`
  ADD PRIMARY KEY (`rackid`);

ALTER TABLE `rack`
  MODIFY `rackid` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;
```

Implement Library System Functionality

First we will create application file `app.py` into project root directory. Then we will import installed module `Flask` and `flask_mysqldb` with it’s helpers. Then we will create Flask app and assign app.config values for making conenction to MySQL database.

```
from flask import Flask, render_template, request, redirect, url_for, session
from flask_mysqldb import MySQL
import MySQLdb.cursors
import re

app = Flask(__name__)

app.secret_key = 'abcd2123445'
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = ''
app.config['MYSQL_DB'] = 'library-system'

mysql = MySQL(app)

@app.route('/')


if __name__ == "__main__":
    app.run()
```

Implement User Login Section

We will create function `login()` in `app.py` and implement login functionality to load login template `login.html` and handle user login functionality to access system.

```
@app.route('/login', methods =['GET', 'POST'])
def login():
    mesage = ''
    if request.method == 'POST' and 'email' in request.form and 'password' in
request.form:
        email = request.form['email']
        password = request.form['password']
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute('SELECT * FROM user WHERE email = % s AND password = % s',
(email, password, ))
        user = cursor.fetchone()
        if user:
            session['loggedin'] = True
            session['userid'] = user['id']
            session['name'] = user['first_name']
            session['email'] = user['email']
            session['role'] = user['role']
            mesage = 'Logged in successfully !'
            return redirect(url_for('dashboard'))
        else:
            mesage = 'Please enter correct email / password !'
    return render_template('login.html', mesage = mesage)
```

we will create login template `login.html` into `templates` directory and create lgoin form html for user login.

```
{% include 'header.html' %}
<body>
  <div class="container-fluid" id="main">
  {% include 'top_menus.html' %}
    <div class="row row-offcanvas row-offcanvas-left">
      <div class="col-md-9 col-lg-10 main">
                <h2>User Login</h2>
                <form action="{{ url_for('login') }}" method="post">
                        {% if mesage is defined and mesage %}
                                <div class="alert alert-warning">{{ mesage }}
</div>
                        {% endif %}
                        <div class="form-group">
                                <label for="email">Email:</label>
                                <input type="email" class="form-control"
id="email" name="email" placeholder="Enter email" name="email">
                        </div>
                        <div class="form-group">
                                <label for="pwd">Password:</label>
                                <input type="password" class="form-control"
id="password" name="password" placeholder="Enter password" name="pswd">
                        </div>
                        <button type="submit" class="btn btn-
primary">Login</button>
                        <p class="bottom">Dont't have an account?  <a
class="bottom" href="{{url_for('register')}}"> Register here</a></p>
                </form>

        <hr>
       </div>
      </div>
    </div>
  </body>
</html>
```

Manage Books

We will create function `books()` in `app.py` to list books. We will get books details from database and load template file `books.html` to list books.

```
@app.route("/books", methods =['GET', 'POST'])
def books():
    if 'loggedin' in session:
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute("SELECT book.bookid, book.picture, book.name, book.status,
book.isbn, book.no_of_copy, book.updated_on, author.name as author_name,
category.name AS category_name, rack.name As rack_name, publisher.name AS
publisher_name FROM book LEFT JOIN author ON author.authorid = book.authorid LEFT
JOIN category ON category.categoryid = book.categoryid LEFT JOIN rack ON
rack.rackid = book.rackid LEFT JOIN publisher ON publisher.publisherid =
book.publisherid")
        books = cursor.fetchall()
        return render_template("books.html", books = books)
    return redirect(url_for('login'))
```

we will create template file `books.html` in `templates` directory and create HTML to list books.

```
{% include 'header.html' %}
<body>
  <div class="container-fluid" id="main">
        {% include 'top_menus.html' %}
    <div class="row row-offcanvas row-offcanvas-left">
          {% include 'left_menus.html' %}
      <div class="col-md-9 col-lg-10 main">
                <h3>Book Listing</h3>
                        <br>
                        <table class="table table-striped">
                        <thead>
                          <tr>
                                <th></th>
                                <th>Book</th>
                                <th>ISBN</th>
                                <th>Author</th>
                                <th>Publisher</th>
                                <th>Category</th>
                                <th>Rack</th>
                                <th>No of copy</th>
                                <th>Status</th>
                                <th>Updated On</th>
                                <th></th>
                                <th></th>
                          </tr>
                        </thead>
                        <tbody>
                          {% for book in books %}
                                  <tr>
                                        <td>
                                        {% if book.picture %}
                                                <img
src="../static/images/{{book.picture}}" width="80" height="90">
                                        {% else %}
                                                <img
src="../static/images/default.jpg" width="80" height="90">
                                        {% endif %}
                                        </td>
                                        <td>{{book.name}}</td>
                                        <td>{{book.isbn}}</td>
                                        <td>{{book.author_name}}</td>
                                        <td>{{book.publisher_name}}</td>
                                        <td>{{book.category_name}}</td>
                                        <td>{{book.rack_name}}</td>
                                        <td>{{book.no_of_copy}}</td>
                                        <td>{{book.status}}</td>
                                        <td>{{book.updated_on}}</td>
                                        <td><a href="{{url_for('edit_book',
bookid=book.bookid)}}" class="btn btn-primary">Edit</a></td>
                                        <td><a href="{{url_for('delete_book',
bookid=book.bookid)}}" class="btn btn-danger">Delete</a></td>
                                  </tr>
                        {% endfor %}
                        </tbody>
                  </table>
        <hr>
```

```
       </div>
      </div>
    </div>
  </body>
</html>
```

Manage Book Issue

We will create function `issue_book()` in `app.py` to list issued books. We will get books details from database and load template file `issue_book.html` to list issued books.

```
@app.route("/issue_book", methods =['GET', 'POST'])
def issue_book():
    if 'loggedin' in session:
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute("SELECT issued_book.issuebookid,
issued_book.issue_date_time, issued_book.expected_return_date,
issued_book.return_date_time, issued_book.status, book.name As book_name,
book.isbn, user.first_name, user.last_name FROM issued_book LEFT JOIN book ON
book.bookid = issued_book.bookid LEFT JOIN user ON user.id = issued_book.userid")
        issue_books = cursor.fetchall()
        return render_template("issue_book.html", issue_books = issue_books)
    return redirect(url_for('login'))
```

we will create template file `issue_book.html` in `templates` directory and create HTML to list issued books.

```
{% include 'header.html' %}
<body>
  <div class="container-fluid" id="main">
        {% include 'top_menus.html' %}
    <div class="row row-offcanvas row-offcanvas-left">
          {% include 'left_menus.html' %}
      <div class="col-md-9 col-lg-10 main">
                <h3>Issued Book</h3>
                        <br>
                        <table class="table table-striped">
                        <thead>
                          <tr>
                                <th>Id</th>
                                <th>Book</th>
                                <th>ISBN</th>
                                <th>User</th>
                                <th>Issue Date</th>
                                <th>Expected Return</th>
                                <th>Return Date</th>
                                <th>Status</th>
                                <th></th>
                                <th></th>
                          </tr>
                        </thead>
                        <tbody>
                          {% for issue_book in issue_books %}
                                  <tr>
                                        <td>{{issue_book.issuebookid}}</td>
                                        <td>{{issue_book.book_name}}</td>
                                        <td>{{issue_book.first_name}}</td>
                                        <td>{{issue_book.issue_date_time}}</td>
                                        <td>{{issue_book.expected_return_date}}
</td>
                                        
                                        <td>{{issue_book.return_date_time}}</td>
                                        <td>{{issue_book.status}}</td>
                                        <td><a href="{{url_for('edit_book',
categoryid=issue_book.issuebookid)}}" class="btn btn-primary">Edit</a></td>
                                        <td><a href="{{url_for('delete_book',
categoryid=issue_book.issuebookid)}}" class="btn btn-danger">Delete</a></td>
                                  </tr>
                        {% endfor %}
                        </tbody>
                  </table>
        <hr>
       </div>
      </div>
    </div>
  </body>
</html>
```

Manage Users

We will create function `users()` in `app.py` to list users. We will get users details from database and load template file `users.html` to list users.

```
@app.route("/users", methods =['GET', 'POST'])
def users():
    if 'loggedin' in session:
        cursor = mysql.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute('SELECT * FROM user')
        users = cursor.fetchall()
        return render_template("users.html", users = users)
    return redirect(url_for('login'))
```

we will create template file `users.html` in `templates` directory and create HTML to list users.

```
{% include 'header.html' %}
<body>
  <div class="container-fluid" id="main">
        {% include 'top_menus.html' %}
    <div class="row row-offcanvas row-offcanvas-left">
          {% include 'left_menus.html' %}
      <div class="col-md-9 col-lg-10 main">
                <h3>User Listing</h3>
                        <br>
                        <table class="table table-striped">
                        <thead>
                          <tr>
                                <th>Name</th>
                                <th>Email</th>
                                <th>Role</th>
                                <th></th>
                                <th></th>
                                <th></th>
                                <th></th>
                          </tr>
                        </thead>
                        <tbody>
                          {% for user in users %}
                          <tr>
                                <td>{{user.first_name}}</td>
                                <td>{{user.email}}</td>
                                <td>{{user.role}}</td>
                                <td><a href="{{url_for('view_user',
userid=user.id)}}" class="btn btn-success">View</a></td>
                                <td><a href="{{url_for('edit_user',
userid=user.id)}}" class="btn btn-primary">Edit</a></td>
                                <td><a href="{{url_for('password_change',
userid=user.id)}}" class="btn btn-warning">Change Password</a></td>
                                <td><a href="{{url_for('delete_user',
userid=user.id)}}" class="btn btn-danger">Delete</a></td>
                          </tr>
                        {% endfor %}
                        </tbody>
                  </table>
        <hr>
       </div>
      </div>
    </div>
  </body>
</html>
```

We are also handling functionality for managing authors, publishers, categories, racks etc. You can download the complete project source code with database dump from below download link.

Download

User Management System with PHP & MySQL Last Updated: 26 Dec , 2022
Datatables Add Edit Delete with Ajax, PHP & MySQL Last Updated: 14 Aug , 2022

Build Helpdesk System with jQuery, PHP & MySQL Last Updated: 14 Jan , 2023
Create Editable Bootstrap Table with PHP & MySQL Last Updated: 19 Sep , 2021
School Management System with PHP & MySQL Last Updated: 02 Jan , 2023
Create Simple REST API with PHP & MySQL Last Updated: 22 Jan , 2023
Build Push Notification System with PHP & MySQL Last Updated: 21 Jan , 2023
Ajax CRUD Operation in CodeIgniter with Example Last Updated: 19 Sep , 2021
Hospital Management System with PHP & MySQL Last Updated: 07 Jan , 2023
Advanced Ajax Pagination with PHP and MySQL Last Updated: 19 Sep , 2021