VIDYAVARDHAKACOLLEGEOF ENGINEERING

(Autonomous, affiliated to VTU)

DEPARTMENTOF

CSE (ARTIFICIALINTELLIGENCE & MACHINELEARNING)

VVCE

Open Ended Experiment Report On

"ONLINE BOOK_STORE PROJECT"

Submitted In partial fulfillment of the requirement for the completion of IV semester of

BACHELOR OF ENGINEERING

Submitted By,

SYED ZIAULLA HUSSAINI	4VV23CI110
SAQLAIN PASHA	4VV24CI409
MOHAMMED ANAS KHAN	4VV24CI406
MOHAMMED SAMI	4VV24CI407

Under the Guidance of,

Prof. Sheeban E Tamanna
Assistant Professor
Dept. of CSE(AI&ML)

CSE (ARTIFICIALINTELLIGENCE & MACHINELEARNING)
2024-25

VIDYAVARDHAKA COLLEGE OF ENGINEERING

DEPARTMENTOF
CSE (ARTIFICIALINTELLIGENCE & MACHINELEARNING)

VVCE

CERTIFICATE

Certified that Assessment Based Activity work entitled "**ONLINE BOOK_STORE PROJECT",** is a bonafide work carried out by **SYED ZIAULLA HUSSAINI(4VV23CI110), SAQLAIN PASHA(4VV24CI409), MOHAMMED ANAS KHAN (4VV24CI406), MOHAMMED SAMI (4VV24CI407)** in partial fulfillment of the requirement for the completion of IV semesters in **AI&ML** of Vidyavardhaka College of Engineering during the year 2024-25. It is certified that all corrections/suggestions indicated for Internal Assessment has been incorporated in the report. The report has been approved as it satisfies the academic requirements with respect to Activity Based Assessment work.

Introduction

The way people buy books has changed significantly in the current digital era. Online platforms are gradually replacing or enhancing traditional bookstores as e-commerce and online services grow in popularity. A web-based tool called the Online Bookstore Management System was created to help with this change by providing a quick and easy way to browse, search, and buy books online.

Using essential functions like viewing book catalogs, managing inventory, placing orders, and managing user roles (admin and customers), this project replicates the essential features of an actual online bookstore. HTML/CSS is used for the frontend, MySQL is used for database administration, and Python (Flask) is used for the backend. The system uses relational database concepts, such as normalization, to guarantee data integrity and seamless operations.

This project not only emphasizes functionality but also showcases practical uses of DBMS principles. It is an exemplary demonstration of how web technologies can be integrated with database systems to produce efficient and robust software solutions for real-world issues.

Abstract

This project aims to develop an Online Bookstore Management System that allows users to browse, search, and purchase books while enabling admins to manage inventory and sales. It utilizes a MySQL relational database to store book, user, and order data. A Flask backend handles routing and logic, and a responsive frontend (HTML/CSS) provides an intuitive interface. The system enforces data integrity through constraints, triggers, and assertions in the database layer.

Objectives

- To build a web-based platform for book browsing and purchasing.
- To manage books, users, and orders efficiently using a relational database.
- To implement features like shopping cart, admin panel, and inventory control.
- To apply DBMS concepts like triggers and assertions in a real-world scenario.

Functionalities Features

- **For Users:**

  - Browse book catalog
  - Add books to cart
  - Place orders

- **For Admin:**

  - Add/Edit/Delete books
  - Manage orders
  - View inventory status

- **Database Triggers & Assertions:**

  - **Trigger 1**: Automatically decrease stock when an order is placed.
  - **Trigger 2**: Prevent placing an order if stock is insufficient.
  - **Assertion 1**: Price must be greater than 0.
  - **Assertion 2**: Stock must not be negative

Technologies Used

➢ Frontend (Client-Side)

- **HTML5**: For structuring web pages and content.
- **CSS3**: For styling the application and improving the user interface.
- **Bootstrap**: For responsive design and UI components.
- **Jinja2**: Templating engine used in Flask to render dynamic content.

➢ Backend (Server-Side)

- **Python 3.x**: Core programming language used for developing server-side logic.
- **Flask**: Lightweight Python web framework used to build and route the application.
- **Flask-MySQLdb**: Flask extension for connecting to MySQL using Python.

➢ Database

- **MySQL**: Relational database used for storing user, book, and order data.
- **SQL**: For writing queries, managing schema, and implementing triggers and assertions.

➢ Other Tools

- **VS Code** : Code editors used for development.
- **XAMPP / MySQL Workbench / phpMyAdmin**: Tools used to manage and interact with the MySQL database.

Advantages and Disadvantages

➢ Advantages

- **24/7 Accessibility**
  - Customers can browse and purchase books anytime, providing convenience and flexibility.
- **Stock Management**
  - Real-time inventory tracking helps avoid overselling and alerts admins when stock is low.
- **User-Friendly Interface**
  - Simple navigation and clear layout make the system easy to use for both customers and administrators.
- **Automation of Processes**
  - Automates tasks like order processing, stock deduction, and record updates, reducing manual workload.
- **Data Integrity**
  - Use of database constraints, assertions, and triggers ensures reliable and consistent data.
- **Scalability**
  - The system can be extended with additional features like user authentication, payment integration, and reviews.

➢ Disadvantages

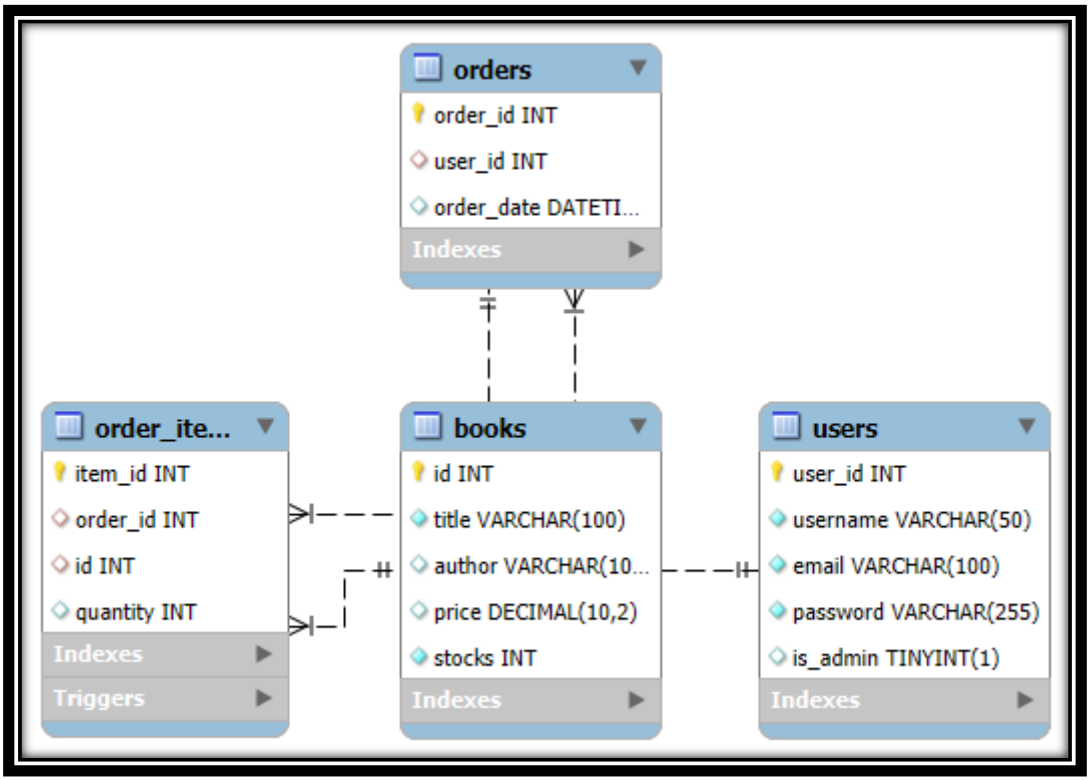
- **Requires Internet Access**
  - Users must have a stable internet connection to access the system, which may be a limitation in remote areas.
- **Security Risks**
  - Without proper implementation, the system can be vulnerable to attacks like SQL injection or unauthorized access.
- **Limited Human Interaction**
  - Lack of in-person recommendations or assistance might reduce the experience for some users.
- **Initial Setup Cost**
  - Hosting, development tools, and database setup may incur a small cost initially if deployed on a live server.
- **Maintenance Needed**
  - Regular updates and database backups are necessary to ensure smooth and secure operation.

Database Design

Tables:

- users (id, name, email, password, role)
- books (id, title, author, price, stock)
- orders (id, user_id, order_date, total_amount)
- order_items (id, order_id, book_id, quantity)

ER Diagram



Source Code

- **app.py**

```
from flask import Flask,flash, render_template, request, redirect, url_for, session
from flask_mysqldb import MySQL
import bcrypt

app = Flask(__name__)
app.secret_key = 'AnaS_05@'

# MySQL Configuration
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = 'root'
app.config['MYSQL_DB'] = 'online_bookstore'

mysql = MySQL(app)

# Sample book data
books = [
    (1, "Book Title 1", "Author 1", 10.99),
    (2, "Book Title 2", "Author 2", 12.99),
    (3, "Book Title 3", "Author 3", 15.99),
]

# Authentication
# Register User
@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        hashed_pw = bcrypt.hashpw(password.encode('utf-8'), bcrypt.gensalt())

        cursor = mysql.connection.cursor()
        cursor.execute("INSERT INTO users (username, email, password) VALUES (%s, %s, %s)",
                       (username, email, hashed_pw))
        mysql.connection.commit()
        return redirect(url_for('login'))
    return render_template('register.html')

# Login User
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']

        cursor = mysql.connection.cursor()
        cursor.execute("SELECT * FROM users WHERE username = %s", (username,))
        user = cursor.fetchone()

        if user and bcrypt.checkpw(password.encode('utf-8'), user[3].encode('utf-8')):
            session['user_id'] = user[0]
            session['is_admin'] = user[4]  # Assuming `is_admin` is the 5th column in the `users` table
            return redirect(url_for('index'))
        else:
            return "Invalid login details"
    return render_template('login.html')
# Logout User
@app.route('/logout')
```

```
def logout():
    session.clear()  # Clear the session to log the user out
    return redirect(url_for('index'))  # Redirect to the homepage

# Displaying books
# Home Page: Display Books
@app.route('/')
def index():
    cursor = mysql.connection.cursor()
    cursor.execute("SELECT * FROM books")
    books = cursor.fetchall()
    return render_template('index.html', books=books)

# Admin Dashboard: Manage Books
@app.route('/admin')
def admin_dashboard():
    if not session.get('is_admin'):
        return "Access denied", 403

    cursor = mysql.connection.cursor()
    cursor.execute("SELECT * FROM books")
    books = cursor.fetchall()
    return render_template('admin_dashboard.html', books=books)

#Adding a new book
@app.route('/add_book', methods=['GET', 'POST'])
def add_book():
    if request.method == 'POST':
        title = request.form['title']
        author = request.form['author']
        price = float(request.form['price'])
        stocks = int(request.form['stocks'])

        cursor = mysql.connection.cursor()
        cursor.execute("INSERT INTO books (title, author, price, stocks) VALUES (%s, %s, %s, %s)",
                       (title, author, price, stocks))
        mysql.connection.commit()
        return redirect(url_for('index'))
    return render_template('add_book.html')

# remove book from database
@app.route('/remove_book/<int:book_id>', methods=['POST'])
def remove_book(book_id):
    if not session.get('is_admin'):
        return "Access denied", 403

    cursor = mysql.connection.cursor()
    cursor.execute("DELETE FROM books WHERE id = %s", (book_id,))
    mysql.connection.commit()
    flash('Book deleted successfully!', 'success')
    return redirect(url_for('admin_dashboard'))

# Cart Management
@app.route('/cart')
def cart():
    if 'cart' not in session or not session['cart']:
        return render_template('cart.html', cart=[], total=0)

    cursor = mysql.connection.cursor()
    cart_books = []
    total = 0
```

```
    for item in session['cart']:
        cursor.execute("SELECT id, title, author, price FROM books WHERE id = %s", (item['book_id'],))
        book = cursor.fetchone()
        if book:
            cart_books.append({
                'book_id': book[0],
                'book_title': book[1],
                'author': book[2],
                'price': book[3],
                'quantity': item['quantity']
            })
            total += book[3] * item['quantity']

    return render_template('cart.html', cart=cart_books, total=total)

# Add to Cart
@app.route('/add_to_cart/<int:book_id>', methods=['POST'])
def add_to_cart(book_id):
    quantity = int(request.form.get('quantity', 1))
    cursor = mysql.connection.cursor()

    # Fetch the current stock for the book
    cursor.execute("SELECT stocks FROM books WHERE id = %s", (book_id,))
    stock = cursor.fetchone()[0]

    # Check if there is enough stock
    if quantity > stock:
        return "Not enough stock available", 400

    # Update the session cart
    if 'cart' not in session:
        session['cart'] = []

    for item in session['cart']:
        if item['book_id'] == book_id:
            if item['quantity'] + quantity > stock:
                return "Not enough stock available", 400
            item['quantity'] += quantity
            break
    else:
        session['cart'].append({'book_id': book_id, 'quantity': quantity})

    # Decrement the stock in the database
    new_stock = stock - quantity
    cursor.execute("UPDATE books SET stocks = %s WHERE id = %s", (new_stock, book_id))
    mysql.connection.commit()

    flash('Book added to cart and stock updated successfully!', 'success')
    return redirect(url_for('index'))

# Remove from Cart
@app.route('/remove_from_cart/<int:book_id>', methods=['GET', 'POST'])
def remove_from_cart(book_id):
    if 'cart' in session:
        session['cart'] = [item for item in session['cart'] if item['book_id'] != book_id]
    flash('Book removed from cart successfully!', 'success')
    return redirect(url_for('cart'))

#Update_Stock
@app.route('/update_stock/<int:book_id>', methods=['POST'])
def update_stock(book_id):
    if not session.get('is_admin'):
```

```
            return "Access denied", 403

        new_stock = int(request.form['new_stock'])
        cursor = mysql.connection.cursor()
        cursor.execute("UPDATE books SET stocks = %s WHERE id = %s", (new_stock, book_id))
        mysql.connection.commit()
        flash('Stock updated successfully!', 'success')
        return redirect(url_for('admin_dashboard'))


    # Checkout & Place Order
    @app.route('/order', methods=['POST'])
    def place_order():
        user_id = session['user_id']
        cart = session.get('cart', [])
        cursor = mysql.connection.cursor()
        cursor.execute("INSERT INTO orders(user_id) VALUES (%s)", (user_id,))
        order_id = cursor.lastrowid

        for item in cart:
            cursor.execute("INSERT INTO order_items(order_id, book_id, quantity) VALUES (%s, %s, %s)",
                    (order_id, item['book_id'], item['quantity']))

        mysql.connection.commit()
        session['cart'] = []
        return redirect(url_for('thank_you'))


    # Thank You Page
    @app.route('/thank_you')
    def thank_you():
        return "Thank you for your order!"


    if __name__ == '__main__':
        app.run(debug=True)
```

- **index.html**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Online Bookstore</title>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
    <!-- Navigation Bar -->
    <nav class="navbar navbar-expand-lg navbar-light bg-light">
        <a class="navbar-brand" href="{{ url_for('index') }}">Online
Bookstore</a>
        <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-
label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
```

```
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav ml-auto">
                    {% if 'user_id' in session %}
                        {% if session.get('is_admin') %}
                            <li class="nav-item">
                                <a class="nav-link" href="{{
    url_for('admin_dashboard') }}">Admin Dashboard</a>
                            </li>
                        {% endif %}
                        <li class="nav-item">
                            <a class="nav-link" href="{{ url_for('cart')
    }}">Cart</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link btn btn-danger text-white" href="{{
    url_for('logout') }}">Logout</a>
                        </li>
                    {% else %}
                        <li class="nav-item">
                            <a class="nav-link" href="{{ url_for('login')
    }}">Login</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link" href="{{ url_for('register')
    }}">Register</a>
                        </li>
                    {% endif %}
                </ul>
            </div>
        </nav>

        <!-- Main Content -->
        <div class="container mt-5">
            <h1>Welcome to Online Bookstore</h1>
            <div class="row">
                {% for book in books %}
                    <div class="col-md-4">
                        <div class="card">
                            <div class="card-body">
                                <h5 class="card-title">{{ book[1] }}</h5>
                                <p class="card-text">Author: {{ book[2] }}</p>
                                <p class="card-text">Price: ${{ book[3] }}</p>
                                <p class="card-text">Stocks: {{ book[4] }}</p>
                                <form action="{{ url_for('add_to_cart',
    book_id=book[0]) }}" method="post">
                                    <input type="number" name="quantity" value="1"
    min="1" max="{{ book[4] }}" class="form-control mb-2">
                                    <button type="submit" class="btn btn-primary"
    {% if book[4] == 0 %}disabled{% endif %}>Add to Cart</button>
```

```
                                </form>
                            </div>
                        </div>
                    </div>
                {% endfor %}
            </div>
        </div>
    </body>
    </html>
```

- **login.html**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login</title>
                                                                    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/css/bootstrap.min.
css" rel="stylesheet">
</head>
<body>
    <div class="container mt-5">
        <h1>Login</h1>
        <form action="{{ url_for('login') }}" method="post">
            <div class="form-group">
                <label for="username">Username</label>
                    <input type="text" class="form-control" id="username"
name="username" required>
            </div>
            <div class="form-group">
                <label for="password">Password</label>
                 <input type="password" class="form-control" id="password"
name="password" required>
            </div>
            <button type="submit" class="btn btn-primary">Login</button>
        </form>
              <p  class="mt-3">Don't  have  an  account?  <a  href="{{
url_for('register') }}">Register here</a>.</p>
    </div>
</body>
</html>
```

- **register.html**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Register</title>
```

```
                                                                    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
    <div class="container mt-5">
        <h1>Register</h1>
        <form action="{{ url_for('register') }}" method="post">
            <div class="form-group">
                <label for="username">Username</label>
                        <input  type="text"  class="form-control"  id="username"
name="username" required>
            </div>
            <div class="form-group">
                <label for="email">Email</label>
                <input type="email" class="form-control" id="email" name="email"
required>
            </div>
            <div class="form-group">
                <label for="password">Password</label>
                     <input  type="password"  class="form-control"  id="password"
name="password" required>
            </div>
            <button type="submit" class="btn btn-primary">Register</button>
        </form>
         <p class="mt-3">Already have an account? <a href="{{ url_for('login')
}}">Login here</a>.</p>
    </div>
</body>
</html>
```

- ## Cart.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Your Cart</title>
                                                                    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/css/bootstrap.min.c
ss" rel="stylesheet">
   <link rel="stylesheet" href="{{ url_for('static', filename='styles.css')
}}">
</head>
<body>
    <div class="container mt-5">
        <h2>Your Cart</h2>

        <!-- Cart Table -->
        <table class="table">
            <thead>
                <tr>
                    <th>Book Title</th>
                    <th>Author</th>
                    <th>Price</th>
                    <th>Quantity</th>
                    <th>Total</th>
                    <th>Action</th>
                </tr>
            </thead>
            <tbody>
                {% for item in cart %}
                    <tr>
                        <td>{{ item.book_title }}</td>
                        <td>{{ item.author }}</td>
                        <td>${{ item.price }}</td>
                        <td>{{ item.quantity }}</td>
                        <td>${{ item.price * item.quantity }}</td>
                               <td><a  href="{{  url_for('remove_from_cart',
book_id=item.book_id) }}" class="btn btn-danger">Remove</a></td>
                    </tr>
                {% endfor %}
            </tbody>
        </table>

        <!-- Checkout Button -->
        <div class="d-flex justify-content-between">
            <h4>Total: ${{ total }}</h4>
                  <a  href="{{  url_for('thank_you')  }}"  class="btn  btn-
primary">Complete Order</a>
```

```
            </div>
        </div>

                        <script          src="https://code.jquery.com/jquery-
3.5.1.slim.min.js"></script>
                                                                        <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/js/bootstrap.bundle.
min.js"></script>
</body>
</html>
```

- **admin.html**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Admin Dashboard</title>
                                                                    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/css/bootstrap.min.c
ss" rel="stylesheet">
    <link rel="stylesheet" href="{{ url_for('static', filename='styles.css')
}}">
</head>
<body>
    <div class="container mt-5">
        <h2>Admin Dashboard</h2>

        <!-- Book Management -->
        <h4>Manage Books</h4>
        <a href="{{ url_for('add_book') }}" class="btn btn-success mb-3">Add
New Book</a>

        <table class="table">
            <thead>
                <tr>
                    <th>Book Title</th>
                    <th>Author</th>
                    <th>Price</th>
                    <th>Stock</th>
                    <th>Action</th>
                </tr>
            </thead>
            <tbody>
                {% for book in books %}
                    <tr>
                        <td>{{ book[1] }}</td>
                        <td>{{ book[2] }}</td>
                        <td>${{ book[3] }}</td>
                        <td>{{ book[4] }}</td>
```

```
                            <td>
                              <a href="{{ url_for('edit_book', book_id=book[0])
    }}" class="btn btn-warning">Edit</a>
                             <a href="{{ url_for('delete_book', book_id=book[0])
    }}" class="btn btn-danger">Delete</a>
                            </td>
                        </tr>
                    {% endfor %}
                </tbody>
            </table>
        </div>

                          <script          src="https://code.jquery.com/jquery-
    3.5.1.slim.min.js"></script>
                                                                          <script
    src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/js/bootstrap.bundle.
    min.js"></script>
    </body>
    </html>
```

- **admin_dashboard.html**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Admin Dashboard</title>
                                                                    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
    <div class="container mt-5">
        <h1>Admin Dashboard</h1>
        <p>Welcome, Admin!</p>
         <a href="{{ url_for('add_book') }}" class="btn btn-primary mb-3">Add
New Book</a>
        <a href="{{ url_for('index') }}" class="btn btn-secondary mb-3">Back to
Home</a>

        <!-- Flash Messages -->
        {% with messages = get_flashed_messages(with_categories=true) %}
            {% if messages %}
                <div class="alert alert-success">
                    {% for category, message in messages %}
                        <p>{{ message }}</p>
                    {% endfor %}
                </div>
            {% endif %}
```

```
            {% endwith %}
            <h2>Manage Books</h2>
            <table class="table table-bordered">
                <thead>
                    <tr>
                        <th>ID</th>
                        <th>Title</th>
                        <th>Author</th>
                        <th>Price</th>
                        <th>Stocks</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>
                    {% for book in books %}
                    <tr>
                        <td>{{ book[0] }}</td>
                        <td>{{ book[1] }}</td>
                        <td>{{ book[2] }}</td>
                        <td>${{ book[3] }}</td>
                        <td>
                          <form action="{{ url_for('update_stock', book_id=book[0])
    }}" method="post" class="form-inline">
                                    <input type="number" name="new_stock" value="{{
    book[4] }}" class="form-control mr-2" min="0">
                                          <button  type="submit"  class="btn  btn-
    success">Update</button>
                             </form>
                        </td>
                        <td>
                           <form action="{{ url_for('remove_book', book_id=book[0])
    }}" method="post" style="display:inline;">
                                          <button  type="submit"  class="btn  btn-
    danger">Remove</button>
                             </form>
                        </td>
                    </tr>
                    {% endfor %}
                </tbody>
            </table>
        </div>
    </body>
    </html>
```

- **addbooks.html**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Add Book</title>
                                                                        <link
href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/css/bootstrap.min.css
" rel="stylesheet">
</head>
<body>
    <div class="container mt-5">
        <h1>Add a New Book</h1>
        <form action="{{ url_for('add_book') }}" method="post">
            <div class="form-group">
                <label for="title">Book Title</label>
               <input type="text" class="form-control" id="title" name="title"
required>
            </div>
            <div class="form-group">
                <label for="author">Author</label>
                        <input  type="text"  class="form-control"  id="author"
name="author" required>
            </div>
            <div class="form-group">
                <label for="price">Price</label>
              <input type="number" step="0.01" class="form-control" id="price"
name="price" required>
            </div>
            <div class="form-group">
                <label for="stocks">Stocks</label>
                      <input  type="number"  class="form-control"  id="stocks"
name="stocks" required>
            </div>
            <button type="submit" class="btn btn-primary">Add Book</button>
        </form>
         <a href="{{ url_for('index') }}" class="btn btn-secondary mt-3">Back
to Home</a>
    </div>
</body>
</html>
```

ONLINE BOOK STORE

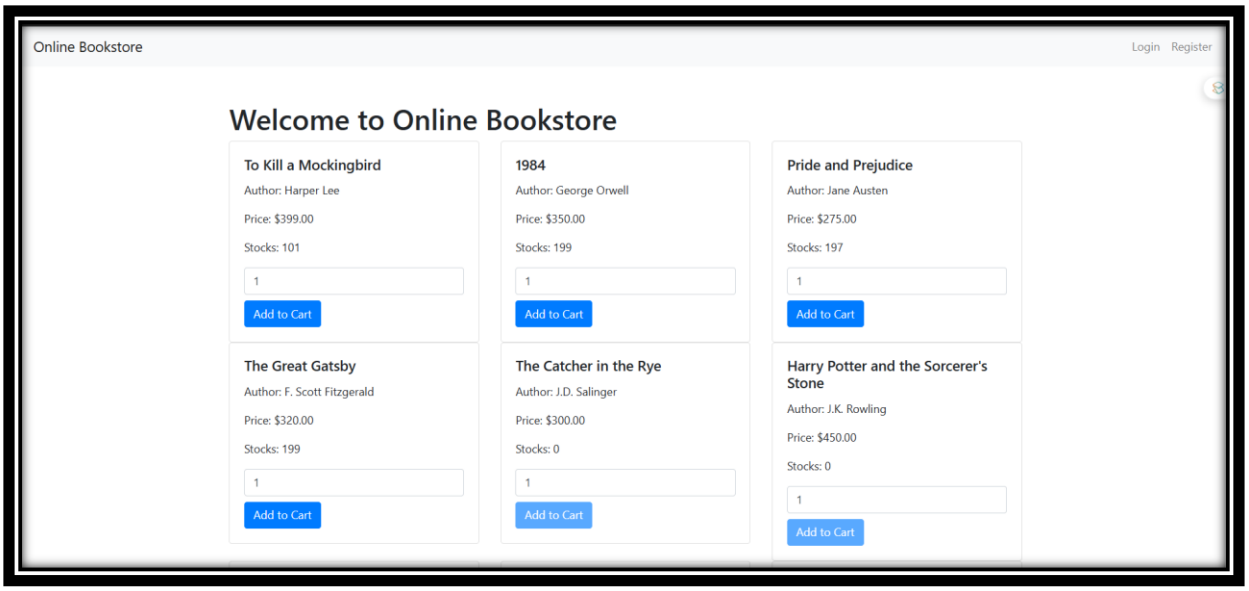
- **mysql.sql**

```
-- 1. Users Table
CREATE TABLE users (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(50) UNIQUE NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    is_admin BOOLEAN DEFAULT FALSE
);

-- 2. Books Table
CREATE TABLE books (
    id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(100) NOT NULL,
    author VARCHAR(100),
    price DECIMAL(10, 2),
    stock INT DEFAULT 0
);

-- 3. Orders Table
CREATE TABLE orders (
    order_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT,
    order_date DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);

-- 4. Order Items Table
CREATE TABLE order_items (
    item_id INT PRIMARY KEY AUTO_INCREMENT,
    order_id INT,
    id INT,
    quantity INT,
    FOREIGN KEY (order_id) REFERENCES orders(order_id),
    FOREIGN KEY (id) REFERENCES books(id)
);

-- Assertions

-- Assertion 1: Book price must be positive
ALTER TABLE books ADD CONSTRAINT price_check CHECK (price > 0);

-- Assertion 2: Quantity of order items must be positive
ALTER TABLE order_items ADD CONSTRAINT quantity_check CHECK (quantity > 0);

-- Triggers
-- DELIMITER $$
-- CREATE TRIGGER decrease_stock
-- AFTER INSERT ON order_items
```

```
-- FOR EACH ROW
-- BEGIN
--   UPDATE books
--   SET stock = stock - NEW.quantity
--   WHERE id = NEW.id;
-- END;$$

-- CREATE TRIGGER prevent_out_of_stock
-- BEFORE INSERT ON order_items
-- FOR EACH ROW
-- BEGIN
--   DECLARE available_stock INT;

--   SELECT stock INTO available_stock
--   FROM books
--   WHERE id = NEW.book_id;

--   IF available_stock < NEW.quantity THEN
--     SIGNAL SQLSTATE '45000'
--     SET MESSAGE_TEXT = 'Not enough stock available';
--   END IF;
-- END;$$
DELIMITER ;
```

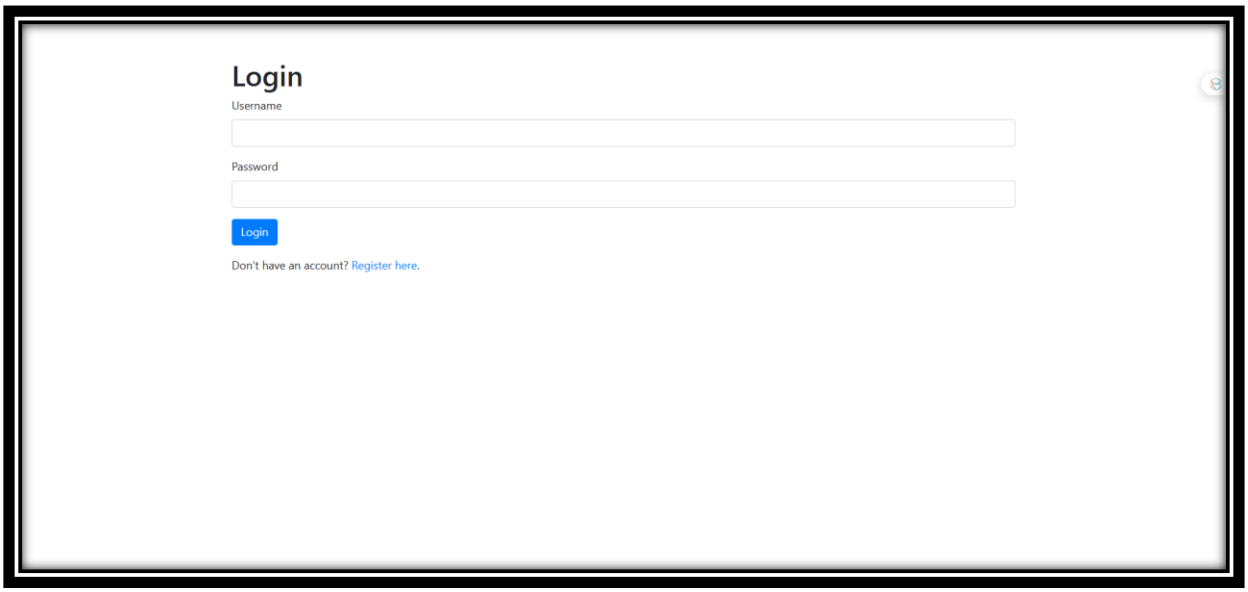
Images

Index.html

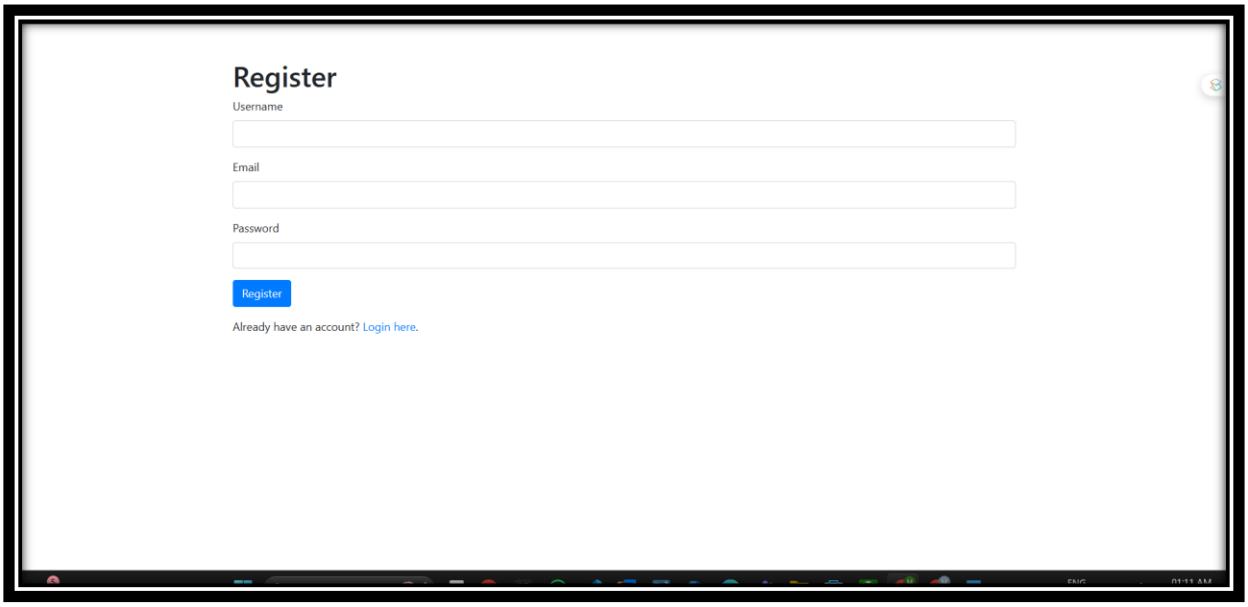


ONLINE BOOK STORE

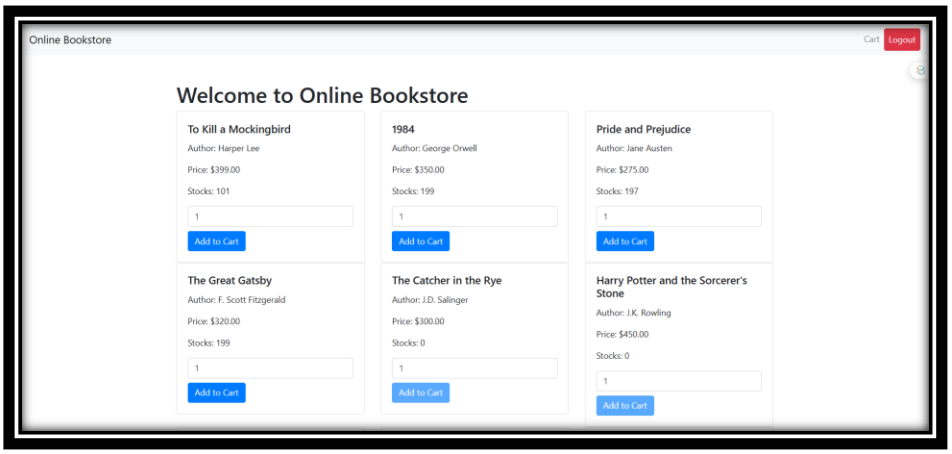
Login.html



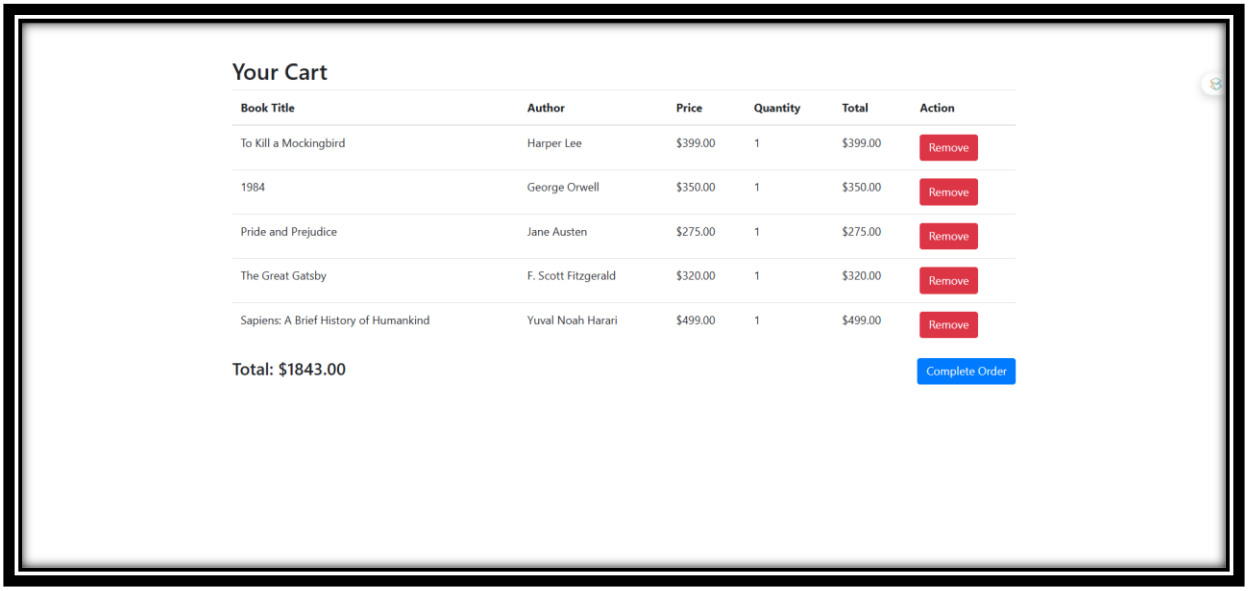
Register.html



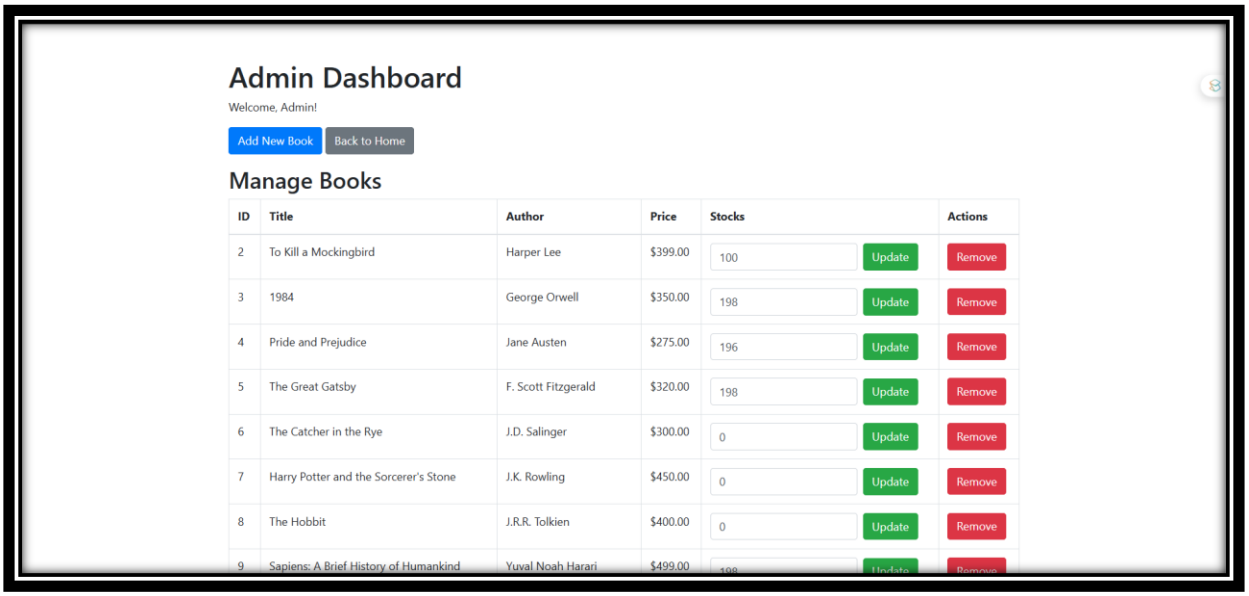
User.html



Cart.html



Admin.html



Conclusion

The Online Bookstore project demonstrates effective use of database management systems in a real-life application. It integrates backend logic, frontend interfaces, and MySQL efficiently while enforcing data consistency and providing a user-friendly experience. This project reinforces practical concepts of DBMS, including DDL, DML, triggers, and constraints.