**1. Project Overview Books to Hunt - Project Documentation**

"Books to Hunt" is a web application built using Flask, SQLite, and HTML/CSS. It allows users to browse, purchase, and manage books in a virtual shopping cart. The project is designed for an online bookstore, where users can view books, add them to their cart, proceed with payment, and check out.

**Key Features:**

* User authentication (Login/Register)
* Product (book) listing with details such as title, author, price, and availability
* Cart functionality to add, update, and view selected books
* Special offers and discounts
* Payment processing with UPI and COD (Cash on Delivery) options
* User-friendly UI with animations and banners

**2. Technologies Used**

* **Flask**: A lightweight Python web framework used to build the server-side logic.
* **SQLite**: A simple relational database used to store user data and product details.
* **HTML/CSS**: Markup and styling for the frontend of the application.
* **JavaScript**: For client-side interactions, such as adding books to the cart, updating quantities, and handling the modal for empty cart checks.
* **QR Code**: Used to generate payment QR codes for UPI transactions.

**Folder Structure**

Books\_to\_Hunt

├── app.py

├── static/

│ └── style.css

├── templates/

├── base.html

├── index.html # Homepage to display books ├── login.html # Login page

├── cart.html

└── order\_summary.html

|── users.db

**4. Functional Modules**

**4.1. User Authentication**

**Login**

* Users can log in with their credentials (username and password).
* If credentials match, the user is redirected to the homepage with a welcome message.

**Register**

* New users can register by providing a username and password.
* User data is stored in the SQLite database, and users are redirected to the homepage after successful registration.

**Session Management**

* Flask sessions are used to store the user's login status and cart details.

**4.2. Product Management (Books)**

* Books are displayed dynamically on the homepage (index.html) by fetching them from the SQLite database.
* Each book has information such as title, author, price, and availability.

**Adding Books to Cart**

* Users can add books to their cart, which is stored in the session. The cart is a dictionary with the book's id, title, price, and quantity.
* The cart data is displayed on the cart.html page.

**4.3. Cart Functionality**

* Users can modify the quantity of items in their cart by increasing or decreasing the quantity using the respective buttons.
* The total price of the cart is updated dynamically.

**4.4. Offers and Discounts**

* The application displays dynamic offers (e.g., "Buy 1 Get 1 Free") on the offers page (offers.html).
* Offers are passed to the template from the server-side.

**4.5. Payment Integration (UPI and COD)**

* The user selects a payment method (UPI or COD) during the checkout (proceed\_to\_pay page).
* For UPI payments, a QR code is generated for the user to scan using their mobile app.

**4.6. Order Summary**

* After payment, the user is shown an order summary (order\_summary.html) with details like total price, expected delivery date, and payment method.
* The expected delivery date is set to 5 days from the current date.

**4.7. Modal for Empty Cart**

* A modal is displayed if the user tries to proceed to payment with an empty cart.JavaScript is used to show and hide the modal dynamically.

**5. Frontend (HTML, CSS, JavaScript)**

**5.1. Base Layout (base.html)**

* The base.html file contains the basic structure of the page (header, navbar, footer) which is extended by other pages like index.html, login.html, etc.
* It uses Jinja templating to dynamically insert data like the user's session information (username).

**5.2. CSS Styling (style.css)**

* The style.css file defines the layout and styling of the application. This includes the navbar, forms, buttons, cart items, and more.
* Animations are also added for certain UI elements like the floating offer banners.

**5.3. JavaScript for Interactivity**

* JavaScript is used to handle the cart's quantity changes and to show the modal for empty cart cases.
* Event listeners are used to switch between the login and registration forms on the login page.

**6. Database Structure (SQLite)**

The database contains two main tables: users and products (books).

**6.1. Users Table**

* Stores user information (username, password).

**6.2. Products Table**

* Stores book details (name, description, price).

**7. Conclusion**

This project is a simple yet powerful web application that demonstrates how to build an online bookstore with functionalities like user login, cart management, and payment options. It is structured to be easily extendable, allowing for the addition of more features in the future, such as order tracking, user profiles, or payment gateway integrations.