

**Project Title**

**“Ecommerce Bookstore”**

|  |  |
| --- | --- |
| Name | Enrollment No. |
| Suchita Shinde | AF0481798 |
| Pranali Kalokhe | AF0481793 |

Under Guidance of

**Rajshri Thete**

**Index**

|  |  |  |
| --- | --- | --- |
| **Sr.no** | **Topic** | **Page no** |
| 1 | Title of Project | 5 |
| 2 | Abstract | 5 |
| 3 | Introduction | 6-9 |
| 4 | Objective | 9- |
| 5 | Programming Language | 9-18 |
| 6 | Data gathering | 19 |
| 7 | System design /UML Diagrams | 20-22 |
| 8 | Proposed system features | 23-25 |
| 9 | Evalution | 26-27 |
| 10 | Future scope | 28 |
| 11 | conclusion | 29 |
| 12 | Biography | 30 |
| 13 | References | 31 |

**Title of Project**

**ECOMMERECE   
BOOKSTORE**

**Abstract**

This project presents the design and development of an **E-commerce Bookstore Web Application** using the Django web framework. The primary objective is to provide users with a convenient platform for browsing, searching, and purchasing books online. The system offers key functionalities such as user registration, login/logout, book listings, detailed book views, shopping cart, checkout, and payment processing via UPI and debit card. The backend is powered by an SQLite database, ensuring lightweight yet effective data management for users, orders, and book inventories.

The bookstore is organized into modular Django apps — including 'books' and 'accounts' — promoting scalability and ease of maintenance. Additionally, data analytics features are integrated to generate insights such as total sales, top-selling books, and customer purchase behavior using Python libraries like Matplotlib and Seaborn.

A recommendation system has been incorporated using both **collaborative filtering** and **content-based filtering** techniques to enhance user experience by suggesting relevant books based on user history and preferences.

This project demonstrates the effective combination of web development, database management, and data science techniques to create a comprehensive and user-friendly online bookstore.

**Acknowledgement**

I take this opportunity to express my profound gratitude and deep regards to my teachers Prof. Rajshri Thete for their exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by them time to time shall carry me a long way in the journey of life on which I am about to embark.

Special gratitude I give to my respected head of the division Mr.XYZ, for allowing me to use the facilities available and also help me to coordinate my project

Furthermore, I would also like to acknowledge with much appreciation the crucial role of faculty members on this occasion.

Lastly, I thank almighty, my parents and friends for their constant encouragement without which this project would not be possible.

1. **Introduction**

In today’s digital-first world, online shopping platforms have revolutionized the way consumers access products and services. Among the wide array of e-commerce solutions, online bookstores provide a convenient and accessible platform for readers to explore, purchase, and review books from the comfort of their homes. This project, titled **"Design and Development of an E-Commerce Bookstore Web Application using Django Framework"**, aims to simulate the features and operations of a real-world online bookstore.

The primary goal is to provide a seamless and secure experience for users to browse through various categories of books, manage their shopping cart, make payments, and receive order confirmations. Simultaneously, the system is designed to aid administrators in managing inventory, monitoring sales, and understanding customer preferences through integrated data analytics.

The bookstore is developed using the **Django web framework**, which supports rapid development and clean, pragmatic design. Django's built-in functionalities for handling user authentication, session management, database ORM, and admin panels make it an ideal choice for developing scalable web applications

1. **Objective**

The objectives of this project are as follows:

* **User-Friendly Interface**  
  Develop an intuitive and responsive user interface that facilitates seamless navigation for customers.
* **Secure User Authentication**  
  Implement secure registration, login, and session management functionalities using Django’s built-in authentication system.
* **Book Management System**  
  Provide comprehensive features for browsing, searching, filtering, and viewing books, including book details such as title, author, genre, price, description, and user ratings.
* **Shopping Cart and Payment Integration**  
  Allow users to add books to a cart, update quantities, and complete purchases using simulated UPI and debit card payment methods.
* **Administrative Control Panel**  
  Enable administrators to manage book inventory, process orders, and monitor overall system activity through a dedicated backend interface.
* **Data Analytics for Business Insights**  
  Incorporate data analysis capabilities to visualize:
* Top-selling books
* Total and monthly revenue
* User purchase patterns
* **Recommendation System**  
  Build a recommendation engine using collaborative and content-based filtering to provide personalized book suggestions based on user behavior and preferences.
* **Scalability and Maintainability**  
  Design the system in a modular, scalable, and maintainable way using the Django MVC architecture and reusable components.

1. **Programming Language**

**1. Python**

Role: Backend development, logic implementation, data processing.

Why Chosen: Python is versatile, easy to read and write, and has a rich ecosystem of libraries for web development, data analysis, and machine learning.

**2. Django (Python Web Framework)**

Role: Main framework used to build the web application.

Key Features Used: Built-in authentication and session management , ORM (Object-Relational Mapping) for database operations, Admin interface for managing data, URL routing, middleware, and template rendering

**3. HTML, CSS, and Bootstrap**

Role: Frontend development for designing and styling the user interface.

Why Chosen: HTML is the standard markup language for structuring web pages.

CSS and Bootstrap provide responsive design and pre-built styling components for faster UI development.

**4. JavaScript**

Role: Enhancing frontend interactivity (e.g., form validation, dynamic content loading).

Why Chosen: JavaScript enables a smoother user experience with client-side scripting.

**5. SQLite**

Role: Database management system for storing user, product, and transaction data.

Why Chosen: Lightweight, serverless, and easy to integrate with Django; suitable for development and testing phases.

**6. Libraries for Data Analysis & Visualization**

matplotlib: For basic plotting and visualizations.

seaborn: For more advanced statistical visualizations.

plotly: For interactive and dynamic charts embedded in dashboards.

**7. Libraries for Recommendation System**

scikit-learn: Used for content-based filtering (e.g., using TF-IDF and cosine similarity).

surprise: Specialized for collaborative filtering models like SVD (Singular Value Decomposition) and KNN (K-Nearest Neighbors).

**8. Development Environment**

VS Code (Visual Studio Code): The primary code editor used during development due to its rich extension ecosystem, Git integration, and debugger.

**3.1 Problem Definition**

The traditional process of purchasing books requires physical presence in bookstores, which may be time-consuming and limited by geographic constraints. With the increasing demand for online services, there is a need for a digital platform that allows users to conveniently browse, search, and purchase books. The core problem is to design a web-based e-commerce system that simplifies the buying process, supports secure transactions, and enhances user engagement through personalized recommendations.

**3.2 Preliminary Investigation**

**Purpose**

The purpose of the preliminary investigation is to evaluate the current challenges faced by book buyers and sellers in the traditional retail space and to explore the possibility of developing an efficient online system.

**Benefits**

* **User Convenience**: Enables 24/7 access to a wide range of books.
* **Wider Reach**: Sellers can reach a global audience.
* **Data Insights**: Analytics help understand user behavior and optimize sales.
* **Cost-effective**: Reduces operational costs compared to physical stores.

**Proposed System**

The proposed system is a Django-based e-commerce bookstore that allows users to:

* Register and manage their profiles.
* Browse, search, and view books by categories, authors, or titles.
* Add books to a cart and checkout using secure payment methods.
* View order history and receive book recommendations.  
  Admins can:
* Add, update, or delete book listings.
* Monitor sales and customer interactions.
* Generate sales reports.

**Feasibility Study**

* **Technical Feasibility**: Developed using Python, Django, and SQLite, all of which are well-supported and widely used.
* **Economic Feasibility**: The project uses open-source tools, minimizing development cost.
* **Operational Feasibility**: The system is user-friendly and meets the needs of both customers and administrators.
* **Schedule Feasibility**: The project plan is achievable within the allotted academic timeline.

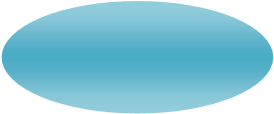
**3.4 Project Planning**

**Purpose of Project Planning**

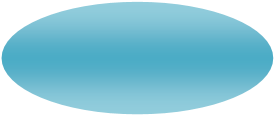
Project planning aims to structure and schedule the development process, ensuring timely delivery, resource optimization, and effective risk management.

**Phases Covered in the Plan**

* **Requirement Gathering and Analysis**  
  Identified system needs, user roles, and key features. Prepared SRS and use case diagrams.
* **System Design**  
  Created UI wireframes, database schema, and system architecture to guide development.
* **Implementation and Coding**  
  Developed key modules like user login, book browsing, cart, checkout, and admin panel using Django.
* **Testing and Debugging**  
  Conducted unit, integration, and system testing to ensure all modules work correctly and the system is bug-free.
* **Deployment**  
  Hosted the project locally or on a server. Configured the environment and migrated the database.
* **Maintenance and Enhancement**  
  Collected feedback, fixed issues, and planned future upgrades like mobile compatibility and real payment integration.



**Stop**



**Start**



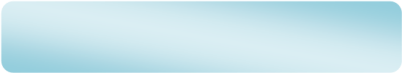
Preliminary



System Analysis



System Design



Coding



Testing



Security



Implementation

**3.5 Software Requirement Specification (SRS)**

**Software & Hardware Requirements**

**Software Requirements:**

* Backend: Python 3.x, Django Framework
* Database: SQLite
* Frontend: HTML, CSS, JavaScript, Bootstrap
* IDE: VS Code
* Libraries: matplotlib, seaborn, scikit-learn, surprise

**Hardware Requirements:**

* Processor: Intel i5
* RAM: 8GB
* Storage:100 GB HDD
* Operating System: Windows11

**3.6 Functional Requirements**

**1. User Management**

* User Registration: Users can create an account by providing name, email, password, and other required details.
* User Login/Logout: Registered users can securely log in and log out of the system.
* Profile Management: Users can view and update their personal details.

**2. Book Browsing and Search**

* Book Listing: Users can browse books by category, author, or latest additions.
* Search Functionality: Users can search for books using keywords like title, author, or genre.
* Book Details Page: Clicking on a book displays detailed information such as description, author, price, and cover image.

**3. Shopping Cart**

* Add to Cart: Users can add books to their shopping cart.
* View Cart: Users can view and modify items in their cart.
* Remove from Cart: Users can remove one or more books before proceeding to checkout.

**4. Checkout and Payment**

* Order Summary: Displays all items in the cart with quantities and prices.
* Address Input: Users enter delivery address details.
* Payment Integration: Users can complete payment via UPI or debit card (simulated).
* Order Confirmation: Displays a confirmation message and order ID upon successful checkout.

**5. Order Management**

* Order History: Users can view their past orders.
* Order Details: Clicking on an order shows the purchased books, payment status, and delivery address.

**6. Admin Panel**

* Admin Login: Only authorized admins can access the admin dashboard.
* Manage Books: Admins can add, update, or delete book entries.
* View Orders: Admins can view all customer orders with details.
* Sales Analytics: Admins can see reports such as total sales, most purchased books, and customer trends.

**7. System design /UML Diagrams**

System design defines the architecture, components, modules, and user interfaces of the project. It transforms functional requirements into a blueprint for building the actual system.

**1. Architectural Design**

The project follows a **three-tier architecture**:

* **Presentation Layer (Frontend)**: HTML, CSS, JavaScript, and Bootstrap are used to design a user-friendly interface.
* **Application Layer (Backend)**: Django framework handles business logic, user requests, routing, and communication between the UI and database.
* **Data Layer (Database)**: SQLite is used to store user details, book data, orders, and transaction records.

**2. Database Design**

A relational database is used with tables such as:

* User: Stores user information (ID, name, email, password).
* Book: Stores book details (ID, title, author, price, stock).
* Order: Records order information (ID, user ID, date, total).
* OrderItem: Links orders with books and quantities.
* Cart: Tracks books added by the user before purchase.

**3. User Interface Design**

* **Home Page**: Displays featured books and categories.
* **Login/Register Page**: For user authentication.
* **Book Detail Page**: Shows book info, "Add to Cart" option.
* **Cart Page**: Shows selected books, prices, and total amount.
* **Checkout Page**: For entering address and selecting payment.
* **Admin Dashboard**: For managing books, viewing sales, and monitoring orders.

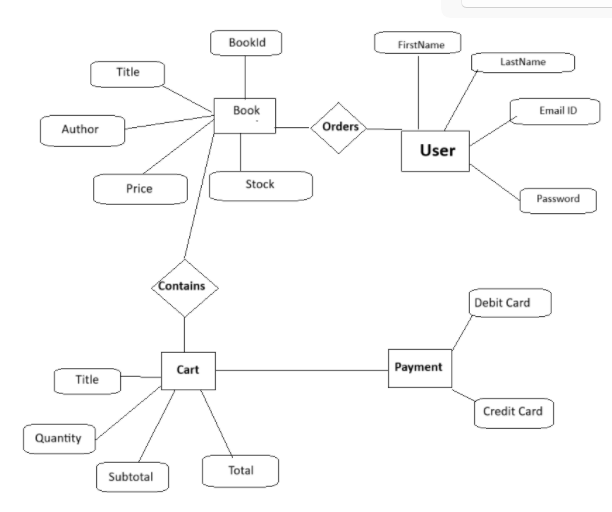
**4. Data Flow Design**

* **User Flow**: User → Browses books → Adds to cart → Checkout → Payment → Order confirmed
* **Admin Flow**: Admin → Login → Manage books/orders → View reports

**5. Technologies Used**

* **Frontend**: HTML, CSS, JavaScript, Bootstrap
* **Backend**: Python, Django
* **Database**: SQLite
* **Analytics & Visualization**: Matplotlib, Seaborn
* **Recommendation System**: Scikit-learn, Surprise library

**ER Diagram**

****

**8. Proposed system features**

The proposed system is a full-featured online bookstore application designed to provide users with a seamless shopping experience and administrators with powerful management tools. The system integrates traditional e-commerce functionalities with advanced data analysis and recommendation features to improve user satisfaction and business performance.

**8.1 User Features**

**1. User Registration and Authentication**

* Secure registration and login system using Django’s built-in authentication module.
* Passwords are securely hashed and stored.
* Role-based access to separate user and admin functionalities.

**2. Book Browsing and Advanced Search**

* Users can browse books by category (e.g., Fiction, Science, Technology).
* Search functionality includes:
  + Keyword-based search (by title, author).
  + Filter by genre, price range, and ratings.

**3. Book Details Page**

* Shows comprehensive details of each book including title, author, price, description, reviews, and cover image.
* Option to rate and review books after purchase.

**4. Shopping Cart and Checkout**

* Add or remove items from the cart.
* Modify item quantity and view dynamic cart totals.
* Checkout with multiple payment options:
  + Simulated UPI
  + Simulated debit card payments

**5. Order History and Tracking**

* Users can view past orders, including order details, payment status, and delivery status.

**8.2 Admin Features**

**1. Admin Dashboard**

* Overview of total users, orders, books, and sales.
* Access to analytical dashboards (monthly revenue, top-selling books).

**2. Book Inventory Management**

* Add, update, or delete book entries from the system.
* Monitor stock levels and receive alerts for low inventory.

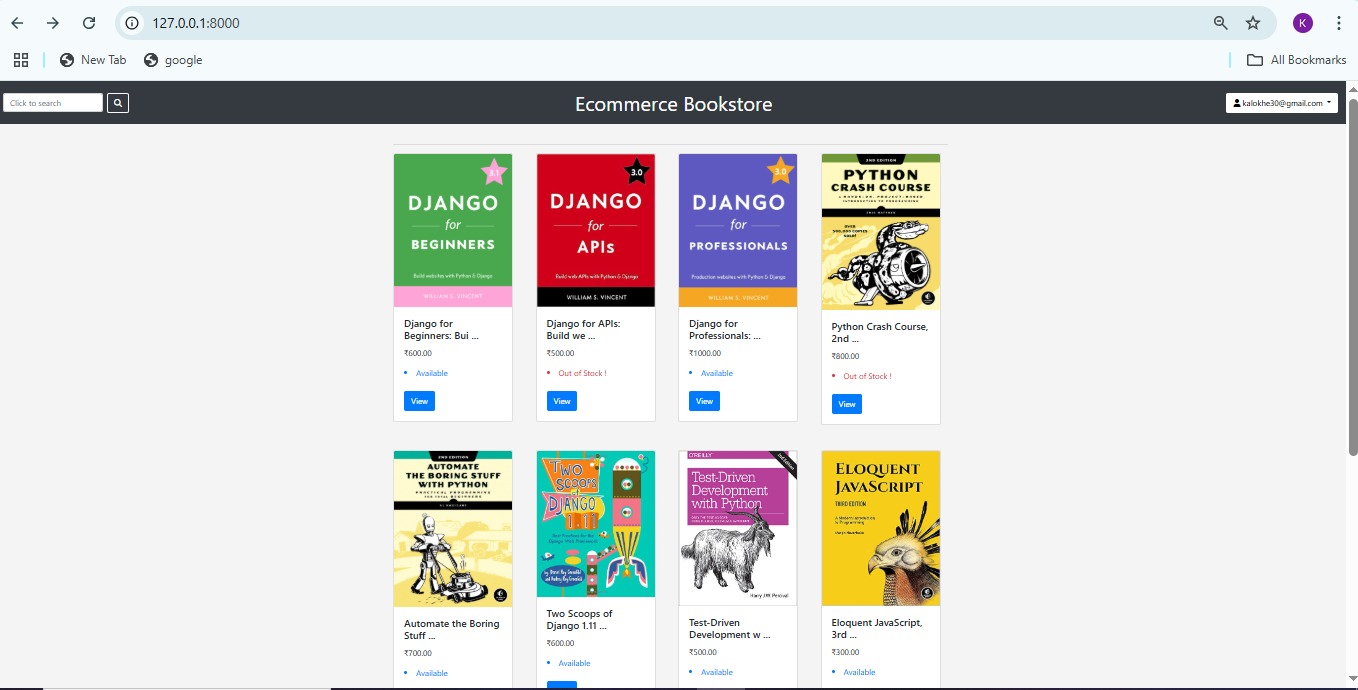
**3. Order Management**

* View and process all customer orders.
* Update order status (pending, completed, shipped, etc.).

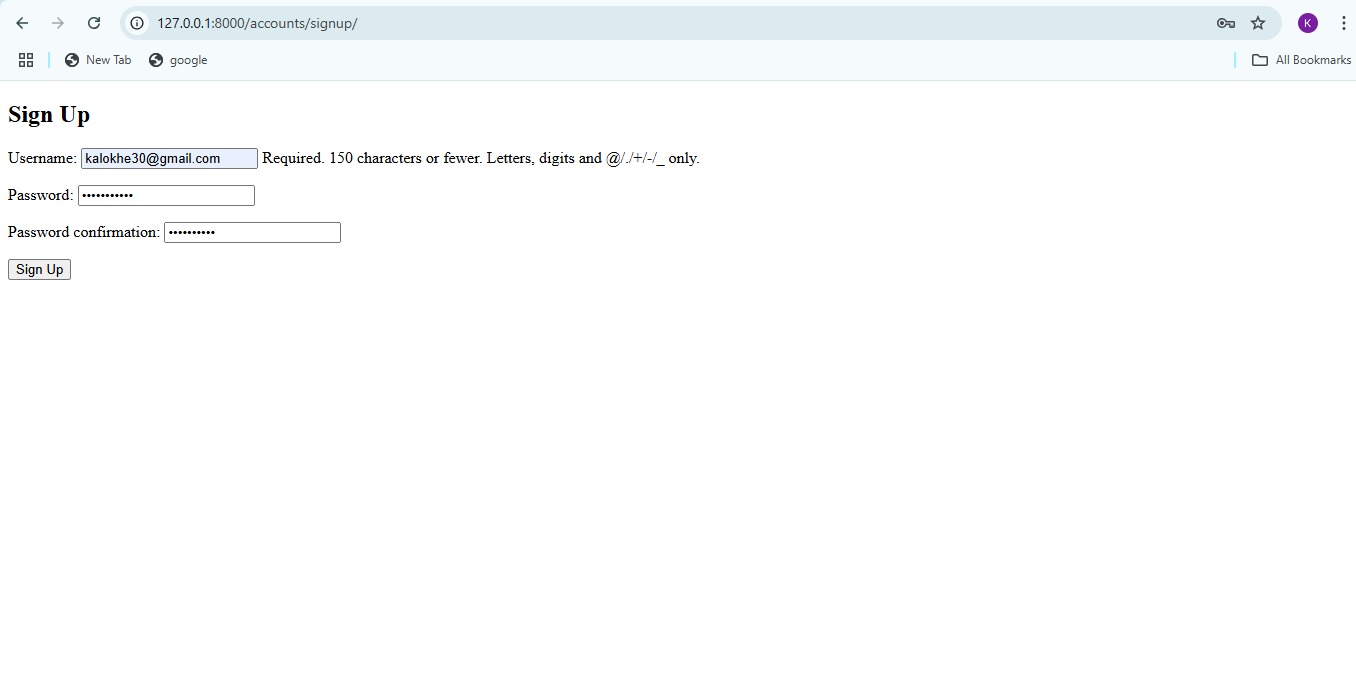
**4. User Management**

* View and manage registered users.
* Assign roles and reset passwords if required.

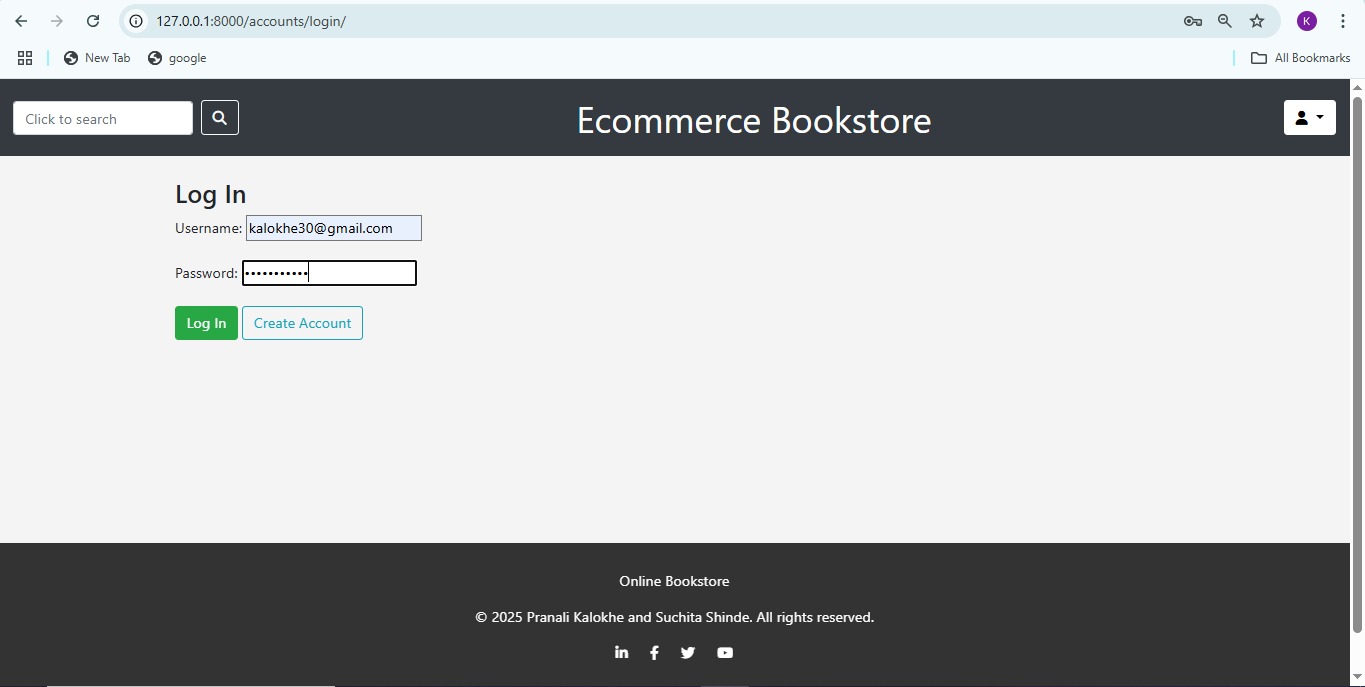
Home page



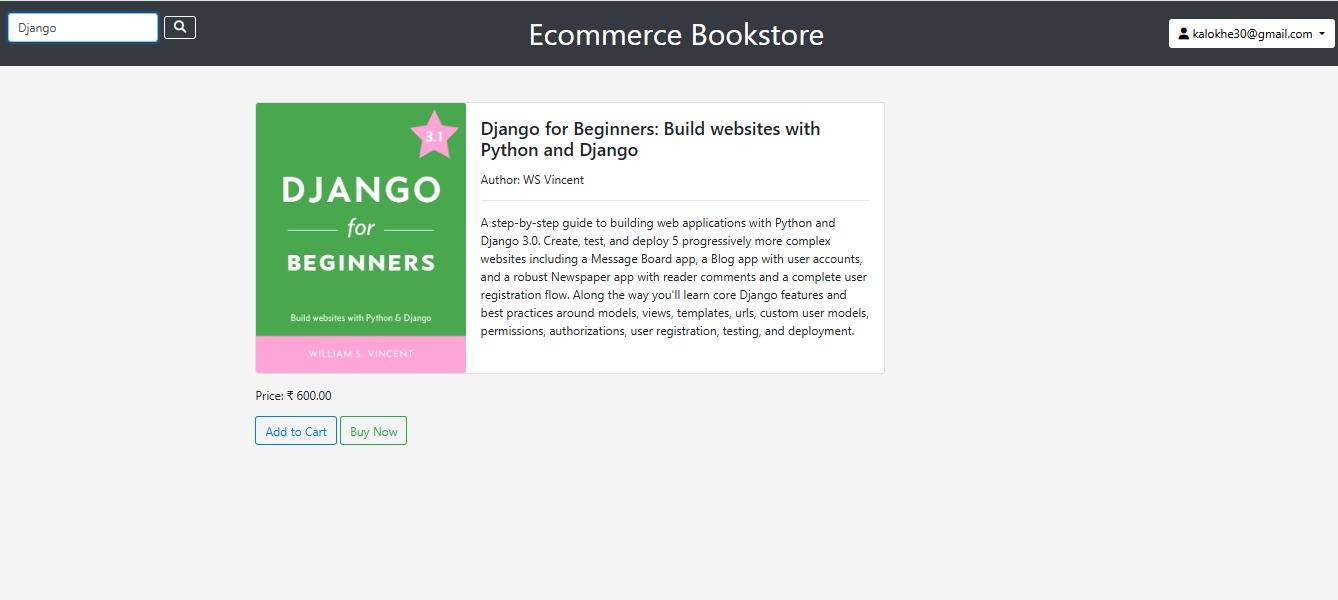
Signup Page



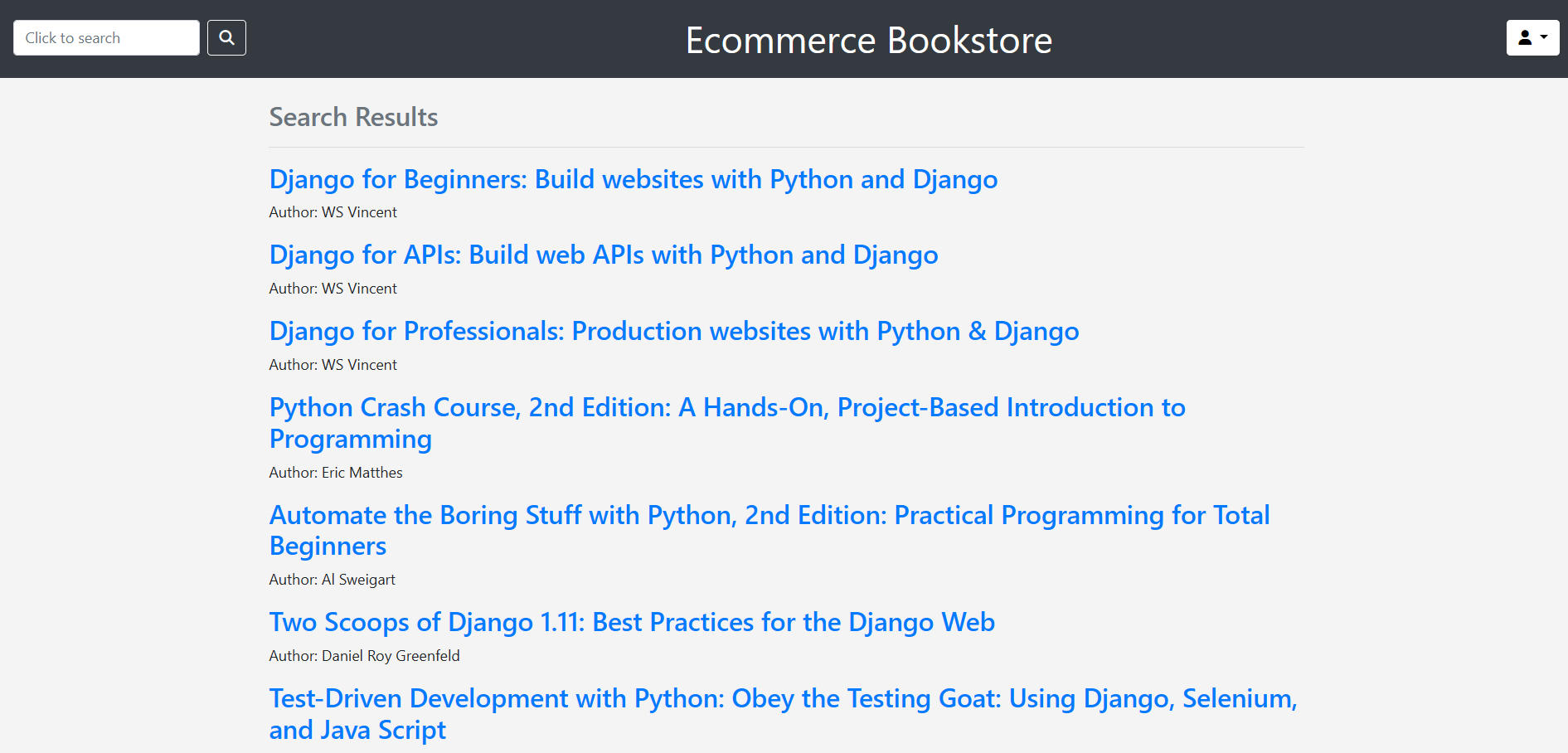
Login Page



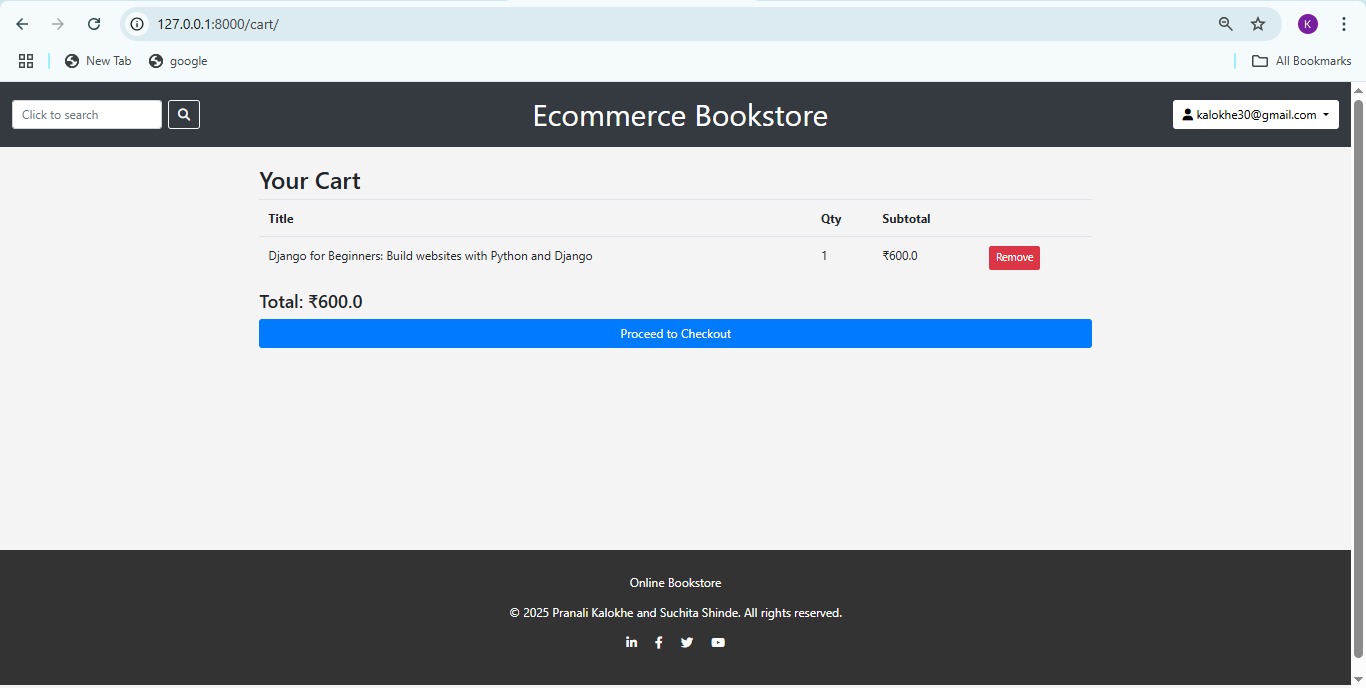
View Page



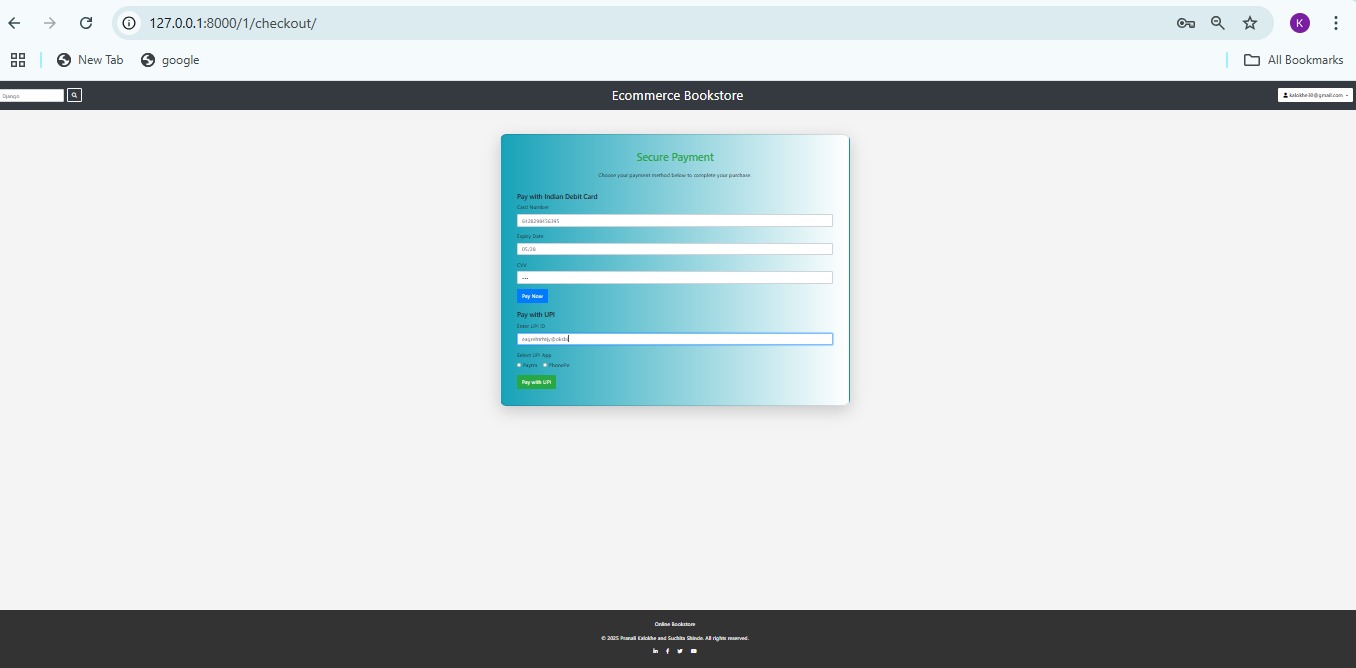
Search Books page



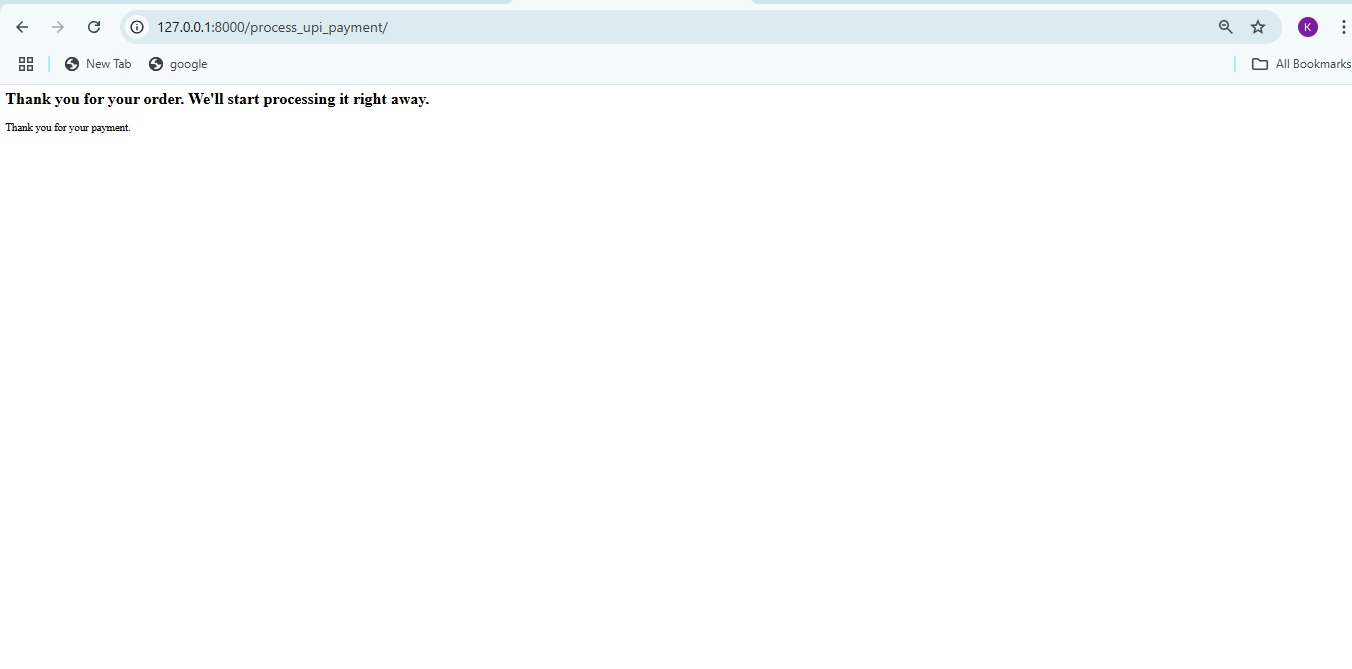
Add to Cart Page



Payment Page



Payment Done Page



**9. Evalution**

Evaluation is essential to determine the effectiveness, usability, and performance of the developed system. This section outlines how the system was assessed through testing, feedback, and performance analysis to ensure it meets user expectations and project objectives.

**9.1 Functional Testing**

All key features of the system were tested to verify that they work as intended. The testing process involved:

* **Unit Testing**: Testing individual components such as user registration, book addition, cart updates, etc.
* **Integration Testing**: Ensuring smooth interaction between modules (e.g., cart → checkout → order confirmation).
* **System Testing**: End-to-end testing of the complete system from user login to payment and order tracking.

|  |  |  |
| --- | --- | --- |
| **Module** | **Test Case Description** | **Result** |
| User Registration | Register with valid/invalid inputs | Pass |
| Login/Logout | Session management and redirection | Pass |
| Book Search | Keyword and filter-based search | Pass |
| Add to Cart | Add/remove items, update quantity | Pass |
| Checkout | Simulate UPI/Debit card transactions | Pass |
| Order Tracking | View order status and history | Pass |
| Admin Book Management | Add/update/delete books via admin panel | Pass |
| Recommendation System | Display personalized book suggestions | Pass |
| Data Visualization | Show graphs on top sellers, revenue, user activity | Pass |

**9.2 Usability Evaluation**

A small group of users (peers and instructors) interacted with the system and provided feedback based on:

* Ease of navigation
* Layout and design
* Loading speed
* Clarity of content
* Overall user experience

**Feedback Summary**:

* The interface was found to be intuitive and responsive.
* Users appreciated the recommendation system and detailed book views.
* Suggestions included: adding a wishlist feature and real-time inventory update notifications.

**9.3 Performance Evaluation**

The system’s performance was tested under different conditions:

* **Load Test**: Simulated 50 concurrent users to check system stability.
* **Database Performance**: Queries were optimized using Django ORM to reduce response time.
* **Page Load Time**: Average page load time was under 2 seconds with static files properly managed.

**9.4 Limitations Identified**

* The current payment system is simulated and not integrated with real-world payment gateways.
* Recommendation accuracy may be limited due to a smaller user base and dataset.
* No mobile app version currently available.

**10. Future scope**

**10.1 Integration with Real Payment Gateways**

* Implement secure and verified payment processing using services like **Razorpay**, **Stripe**, or **PayPal** to handle real transactions instead of simulated ones.
* Include payment confirmation via email/SMS notifications.

**10.2 Mobile Application Development**

* Develop an Android and iOS mobile app using **Flutter** or **React Native** for better accessibility.
* Synchronize web and mobile data through a common backend API.

**10.3 Advanced Recommendation System**

* Enhance collaborative filtering with deep learning models (e.g., neural collaborative filtering).
* Incorporate **hybrid models** that combine user ratings, browsing behavior, and item metadata.
* Include real-time recommendation updates.

**10.4 Enhanced Admin Analytics Dashboard**

* Introduce drill-down dashboards using tools like **Power BI** or **Tableau** integrated via APIs.
* Real-time sales tracking, customer retention analytics, and inventory forecasting.

**10.5 Real-Time Inventory Management**

* Automatically update book availability with real-time stock alerts for low inventory.
* Integrate supplier APIs for automatic restocking.

**10.6 Multi-Vendor Support**

* Expand the platform to allow multiple sellers to list and manage their own books.
* Include seller ratings and commission-based revenue sharing models.

**10.7 Chatbot for Customer Support**

* Add an AI-powered chatbot using **Dialogflow** or **Rasa** to assist users with FAQs, order tracking, and book suggestions.

**10.8 Wishlist and Notifications**

* Enable users to add books to a wishlist and receive notifications about price drops or availability.
* Notify users about new arrivals in their favorite genres.

**11. Conclusion**

This Django-based e-commerce bookstore project successfully demonstrates how to build a functional and user-friendly online shopping platform with integrated data science features. Core functionalities like book browsing, cart, checkout, and admin management were effectively implemented. Additionally, recommendation systems and analytics dashboards enhanced the platform’s intelligence and user experience.

The project strengthened skills in full-stack development, machine learning, and data visualization. Though fully functional, future improvements like real payment integration and mobile support could make the system production-ready. Overall, the project achieved its goals and provided valuable hands-on learning.

**12. BIBLIOGRAPHY**

 **Django Documentation** – <https://docs.djangoproject.com/>

 **SQLite Documentation** – https://www.sqlite.org/docs.html

 Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media.

 McKinney, W. (2022). *Python for Data Analysis*. O'Reilly Media.

 Raschka, S., & Mirjalili, V. (2019). *Python Machine Learning*. Packt Publishing.

 Pedregosa, F., et al. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830.

 Surprise Library – <https://surprise.readthedocs.io/>

 Seaborn Documentation – https://seaborn.pydata.org/

 Plotly Documentation – https://plotly.com/python/

**13. REFERENCES**

· Django Homepage – <http://www.djangoproject.com/>

· Python Documentation – <http://www.python.org/doc>

· Django (Web Framework) – <http://en.wikipedia.org/wiki/Django>

· Django Documentation – <http://docs.djangoproject.com>

· Python (Programming Language) – <http://en.wikipedia.org/wiki/Python>

· MySQL Official Site – <https://www.mysql.com/>

· MySQL Tutorials – <http://www.mysqltutorial.org>

XAMPP Download & Documentation – <https://www.apachefriends.org/download.html>