IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

# ONLINE ELECTRONIC E-COMMERCE PLATFORM

**Govardhini Lakshmi Vakada, R.Rajesh**
Assistant professor, MCA Final Semester, Master of Computer Applications, Sanketika Vidya Parishad Engineering College, Vishakhapatnam,
Andhra Pradesh, India

***Abstract:*** This project report explains how I designed, built, and launched an e-commerce website using the Django framework. The main goal of this website is to give both customers and admins a smooth, secure, and easy-to-use online shopping experience. Customers can browse different products, add them to their cart or wish list, and place orders easily. The website includes safe login options like regular sign-up and sign-in using Google, Facebook, and Apple, which makes it more user-friendly. One of the special features of this project is the multi-vendor system. This means many sellers can sign up, manage their profiles, and upload products in different categories. The website also includes features like product categories, discount vouchers, a blog section, and a wish list to make shopping more personalized. The admin dashboard uses the Jazzmin theme, which gives a clean and simple interface to manage products, orders, users, vendors,
and blog content. The backend is built to make sure data is safe, orders are processed quickly, and stock is managed in real-time.
Security features like input checks, user permissions, and safe handling of data are followed. The project is also designed to grow in the future using reusable parts and helpful third-party tools from Django. All parts of the system were tested to make sure everything works well. This project shows how different technologies and open-source tools can be combined to build a real-world online shopping website.

***Index Terms*** **- Django framework, Visual Studio Code, PyCharm, Sublime Text, PostgreSQL, MySQL**.

## 1. INTRODUCTION

The film industry is a highly dynamic and economically significant sector, where the success of a movie can determine the profitability of producers, studios, and investors [8]. Predicting a movie's box office revenue before its release is a challenging task influenced by multiple variables, including budget, genre, cast, and release timing. Traditionally, such predictions have relied on expert opinions, historical trends, or intuition, often leading to unreliable results due to the complex interplay of factors involved. With the advent of data science and machine learning (ML), it is now possible to analyse large volumes of structured and unstructured data to uncover hidden patterns that influence a movie's financial outcome [4]. Machine learning models can learn from historical data and generate predictions by identifying relationships between input features and revenue [3]. This enables studios and analysts to make more informed and data-driven decisions, reducing financial risk and optimizing marketing strategies This project focuses on the application of machine learning techniques—specifically, Linear Regression, Random Forest, and XGBoost—to forecast movie revenues [5]. A dataset containing features such as movie budget, runtime, genre, and release date is used for training and evaluation [9]. After data cleaning and Pre-processing, the models are assessed using metrics like Mean Absolute Error (MAE) and $R^2$ score [2]. The goal is to determine which model performs best in terms of accuracy and reliability, demonstrating the value of ML in box office prediction [12].

### 1.1 EXISTING SYSTEM

In today's digital world, online shopping has completely changed how people and businesses buy and sell products. Thanks to faster internet and advanced web technologies, shopping online is no longer just a trend—it's a daily part of life for millions of people around the world.

Whether it's clothes, electronics, food, or services, e-commerce websites make it possible to shop from home at any time of day. These platforms have become a major part of the global economy, helping even small businesses reach customers far away.
This project focuses on building a strong and feature-rich e-commerce website using the Django framework, a popular web development tool in Python. Django is known for its security, speed, and flexibility, making it a great choice for building modern websites that are both powerful and easy to manage. The main goal of this project is to create an online shopping platform that gives a smooth, easy, and safe experience for both customers and sellers (vendors).

### 1.1.1 CHALLENGES

The rapid growth of online shopping has exposed the limitations of traditional retail and basic e-commerce platforms [14]. Many existing systems lack advanced features, efficient vendor management, and user engagement tools, resulting in a sub optimal experience for both.

## 1.2 PROPOSED SYSTEM

The proposed system is a next-generation e-commerce web application designed to address the limitations of existing online shopping solutions and provide a seamless, efficient, and engaging experience for both customers and vendors. Developed using the Django framework, the system leverages modern web technologies and best practices to ensure reliability, scalability, and security

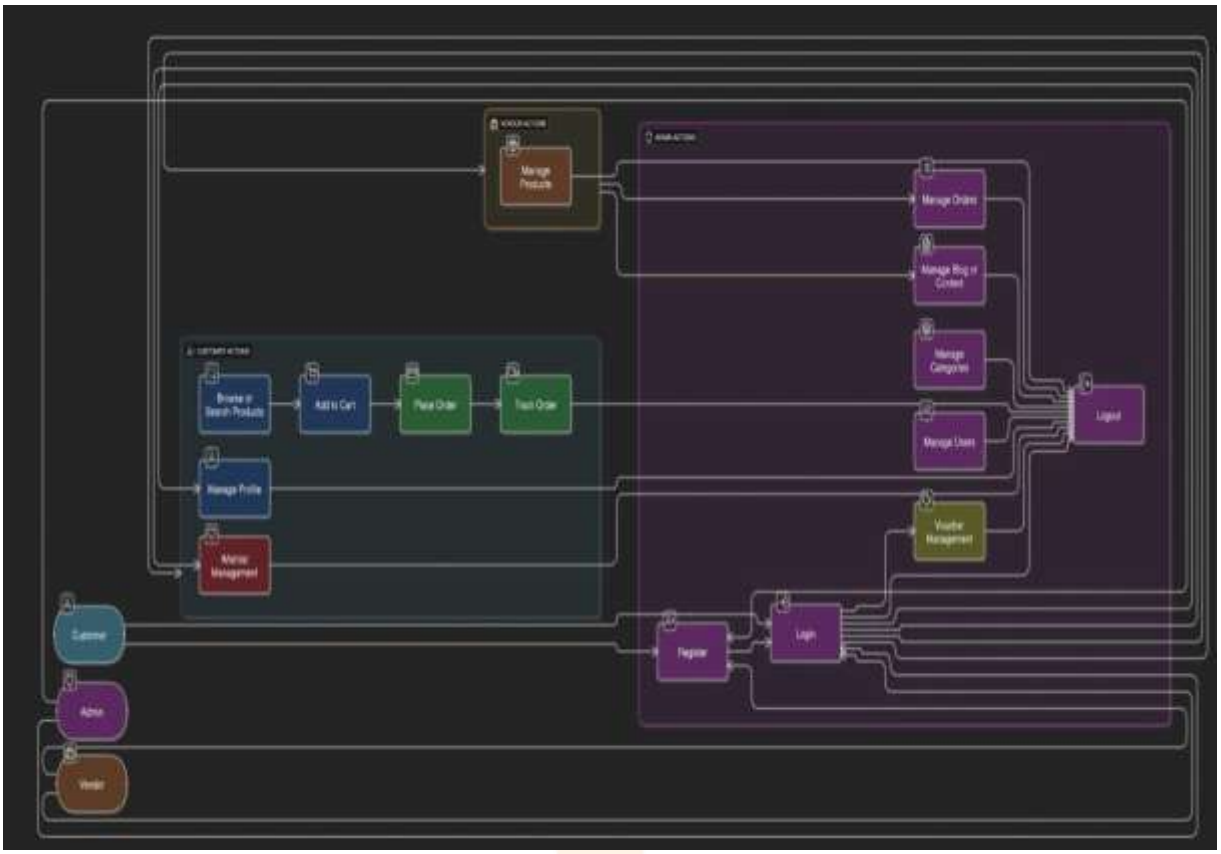


**Fig. 1** Use Case Diagram

### 1.2.1 ADVANTAGES

Enhanced User Experience

- Intuitive and user-friendly interface with advanced product search, filtering, wishlist's, and vouchers [8].
- Seamless shopping cart and order tracking system for convenience [12].
- Responsive design that works smoothly across **desktops, tablets, and mobiles [2] .**

**2. Scalability and Extensibility**

- Modular architecture allows easy integration of new features and supports platform growth as the user base and product catalog expand [12].
- Django's MVT architecture ensures maintainability and adaptability for future enhancements [1].

**3. Efficient Vendor and Product Management**

- Dedicated vendor modules simplify product listings, inventory updates, and categorization [7].
- Admin dashboard (enhanced with Jazzmin) improves backend management of users, products, and content [3].

**4. Community Engagement**

- Integrated blog and content management system helps build user loyalty and increases engagement [6].

**5. Security and Data Privacy**

- Django's built-in security measures protect against common web vulnerabilities [7].
- Strong authentication, including social logins, and secure data handling [23].

**6. Cost-Effectiveness**

- Use of open-source technologies minimizes licensing and development costs [11].
- Efficient backend structure reduces future maintenance costs [21].

**7. Reliability and Performance**

- Robust database and efficient data management ensure fast page loads and smooth performance [3].
- Thorough testing (unit, integration, and manual) increases reliability and user satisfaction [22].

**8. Competitive Advantage**

- Combines modern e-commerce features (multi-vendor support, vouchers, wish list, blog) that many existing platforms lack, making it a next-generation solution [17].

## 2. LITERATURE REVIEW

E-commerce has evolved into a critical component of modern retail, with online platforms reshaping the way consumers and businesses interact [22]. According to Laudon and Traver (2021), the growth of internet technologies and secure payment systems has made online shopping a mainstream activity worldwide [12]. Modern e-commerce solutions aim to provide seamless, secure, and personalized experiences to users, while also offering scalable and efficient management tools for vendors and administrators [21].

Traditional e-commerce systems often suffer from limitations such as inadequate authentication options, poor user interfaces, and inefficient vendor management processes [14]. Many lack advanced features like wishlist's, voucher systems, and integrated content management, which are now considered essential for user engagement and retention [18]. The increasing demand for personalized user experiences has also led to the adoption of advanced search, filtering, and recommendation engines [13].

### 2.1 ARCHITECTURE

The application is built using Django's Model-View-Template (MVT) architecture, which ensures a clear separation of concerns:

- **Model:** Manages the data and business logic, interacting with the database to store and retrieve information about users, products, orders, vendors, etc [5].
- **View:** Handles the processing of user requests, applies business logic, and communicates with models to fetch or update data [17].
- **Template:** Renders the data into HTML, providing a dynamic and user friendly interface [19]. The system is modular, allowing for easy maintenance, testing, and future enhancements.

• **Model Training (Linear Regression) Model:** Manages the data and business logic, interacting with the database to store and retrieve information about users, products, orders, vendors, etc. **View:** Handles the processing of user requests, applies business logic, and communicates with models to fetch or update data [7].

**Template:** Renders the data into HTML, providing a dynamic and user friendly interface [18]. The system is modular, allowing for easy maintenance, testing, and future enhancements [16].

**Module Design**

**User Authentication & Authorization**

- Supports user registration, login, logout, password reset, and social authentication (Google, Facebook, Apple) [9].
- Role-based access control for customers, vendors, and administrators [22].

**Product & Category Management**

Vendors and admins can add, update, and delete Products are organized into categories and subcategories for easy navigation. Product details include images, descriptions, tags, and vendor information [14].

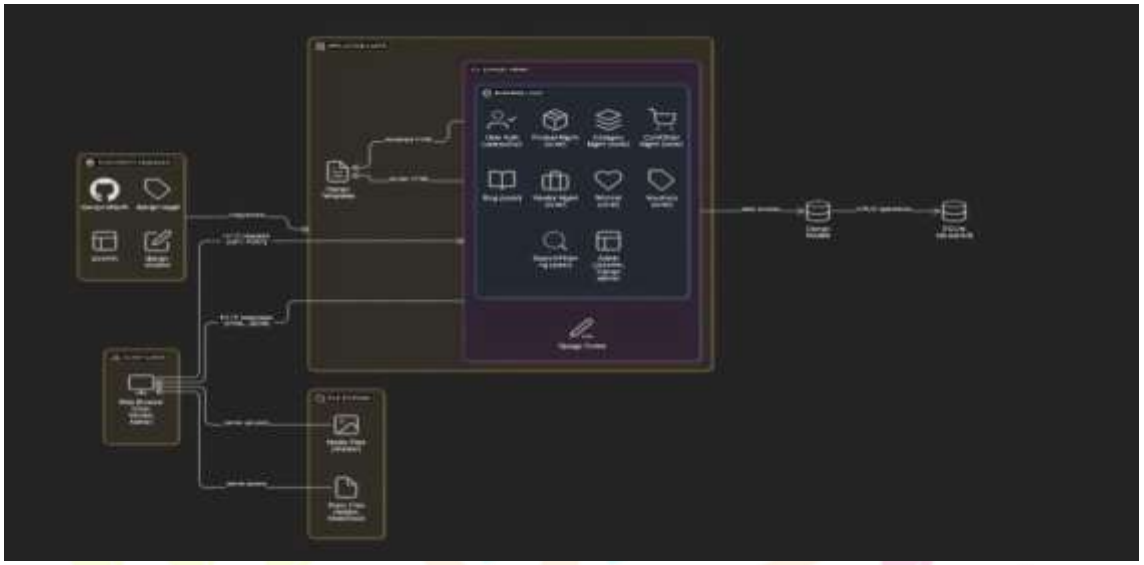

**Fig. 2** System Architecture of Online Electronics E-commerce Platform

## 2.2 ALGORITHM

This algorithm outlines the steps from adding a product to the cart through successful order placement and voucher application [18].
Algorithm Name: Online_Product_Purchase_Workflow
Input:

- User Credentials (username/password or social login)
- Product Selection (Product_ID)
- Quantity
- Voucher Code (optional)
- ShippingAddress Output:
- Order Confirmation
- Order ID
- Total Amount (after voucher)

## 2.3 TECHNIQUES

In this project, various modern web development techniques were used to build a scalable and secure e-commerce platform [19]. The Django framework with the Model-View-Template (MVT) architecture was employed for clean separation of logic and presentation [2]. User authentication was implemented using both traditional and social login methods via django-allauth, along with role-based access control for customers, vendors, and admins. The relational database was designed using Django ORM with normalization up to 3NF to ensure data integrity [12]. Security measures such as CSRF protection, input validation, and password hashing were followed. Responsive design ensured compatibility across devices [18]. Features like product search and filtering, dynamic voucher application, and vendor dashboards were implemented for better usability [11]. The admin panel was enhanced using Jazzmin, and rich content management was supported using django-ckeditor. Static and media file handling, along with thorough testing, ensured smooth and maintainable deployment [8].

## 2.4 TOOLS

The project was developed using the Django framework (Python) as the core backend technology. SQLite was used as the development database, with the option to upgrade to PostgreSQL or MySQL for production [11]. For frontend development, HTML5, CSS3, and JavaScript were used along with Django templates. Visual Studio Code served as the primary code editor [22]. Key third-party packages included django-allauth for authentication, django-ckeditor for rich text editing in blog posts, django-taggit for tagging functionality, and Jazzmin for an enhanced admin interface [16]. Git was used for version control, and pip for managing Python dependencies. Static and media files were handled using Django's built-in file management system, and testing was performed using Django's built-in testing tools [15].

## 2.5 METHODS

The development of the e-commerce platform followed a modular and iterative development methodology, where each feature was implemented and tested in phases [7]. The Model-View-Template (MVT) pattern in Django was the core method used to structure the application, ensuring separation of concerns [12]. Role-based access control was used to manage user permissions for customers, vendors, and administrators [18]. CRUD operations (Create, Read, Update, Delete) were applied throughout for managing products, orders, blogs, and users. Form handling methods in Django were used for validating and processing user inputs securely. Session management was applied for cart handling and login states [9]. Voucher validation was implemented using logical checks on date ranges and usage status [3]. Additionally, object-oriented programming (OOP) principles were followed to structure models and views efficiently, and unit testing methods were used to verify functionality and ensure reliability

## 2.6 METHODOLOGY

The development of the Online Electronics E-commerce Platform followed a modular and iterative Software Development Life Cycle (SDLC) approach [7], ensuring continuous improvement, scalability, and user satisfaction. The methodology can be broken down into the following phases**:**

1. **Requirement Analysis**

- Conducted surveys, interviews, and research on existing e-commerce platforms [9].
- Identified functional requirements (user registration, product catalog, shopping cart, order tracking, etc.) and non-functional requirements (security, scalability, and performance) [7].
- Studied limitations of existing systems to design a feature-rich platform with multi-vendor support and advanced modules [13].

2. **Feasibility Study**

- Technical Feasibility: Evaluated Django as the core framework for its security, modularity, and scalability [11].
- Economic Feasibility: Leveraged open-source tools to reduce development costs [22].
- Operational Feasibility: Ensured that the platform would be user-friendly for customers, vendors, and administrators [23].
- Schedule Feasibility: Created a timeline with milestones for design, coding, and testing phases [19].

3. **System Design**

- Designed the architecture using Django's Model-View-Template (MVT) structure for clear separation of concerns [4].
- Prepared Data Flow Diagrams (DFD), UML diagrams (Use Case and Class Diagrams), and a Database ER diagram [13].
- Designed responsive templates for a seamless user experience across devices [14].

4. **Implementation**

- Developed modular Django apps for user authentication, product and category management, shopping cart, orders, vendor dashboards, blog, vouchers, and wishlist [6].
- Integrated third-party packages such as django-allauth for social authentication and Jazzmin for a modern admin interface [22].
- Implemented robust backend logic for inventory updates, order processing, and vendor management [21].

5. **Testing**

- Performed unit testing, integration testing, and manual validation of all modules [11].
- Verified functionality of key workflows like registration, cart operations, checkout, order tracking, and admin tasks [22].
- Conducted security testing to prevent vulnerabilities like XSS and SQL injection [1].

6. **Deployment**

- Configured production settings with secure keys, database migrations, and static file management.
- Optimized the platform for speed, performance, and scalability [22].

7. **Maintenance and Future Enhancements**

- Designed the system to be easily extendable with upcoming features such as payment gateways, real-time notifications, and AI-

based product recommendations [20].

- Planned regular updates for security patches and performance improvements [13].

### 3.1 INPUT

The input for the predictive modelling system includes key attributes that influence a movie's box office performance and are available prior to its release [5]. These inputs are: budget, which indicates the total production cost; runtime, representing the duration of the film in minutes; genre, a categorical feature that classifies the movie type (e.g., Action, Drama); release date, which can be used to derive seasonal or holiday timing advantages; and production company, which may reflect the brand reputation and previous success rates [17]. These features are selected for their relevance and impact on revenue and are transformed through preprocessing to ensure compatibility with machine learning models [22]**.**


Fig. 3 Predictive Modelling Of Movie Revenue from Kaggle Datasets

### 3.2 METHOD OF PROCESS

**Process model used:** Iterative / modular SDLC on top of Django's MVT architecture, with continuous testing and scope for future enhancements [5].

**Dimensions covered:**

- **Technical:** Django, django-allauth, Jazzmin, Taggit, CKEditor, SQLite → PostgreSQL/MySQL ready [8].
- **Operational:** Low learning curve, role-based access, clean admin [1].
- **Economic:** Open-source stack → low licensing & maintenance cost.
- **Legal & Social:** Data privacy, compliance, fair marketplace practices [4].
- **Schedule:** Milestone-based plan enabling parallel module development.


**Fig. 4 User Login Page**


**Fig. 5 Blog Selection Page**

## 4. RESULTS

The Django e-commerce platform, as developed, is robust, secure, and user friendly [7]. It meets the project's objectives and provides a solid foundation for future growth. The use of modern tools like Jazzmin for the admin interface [17],

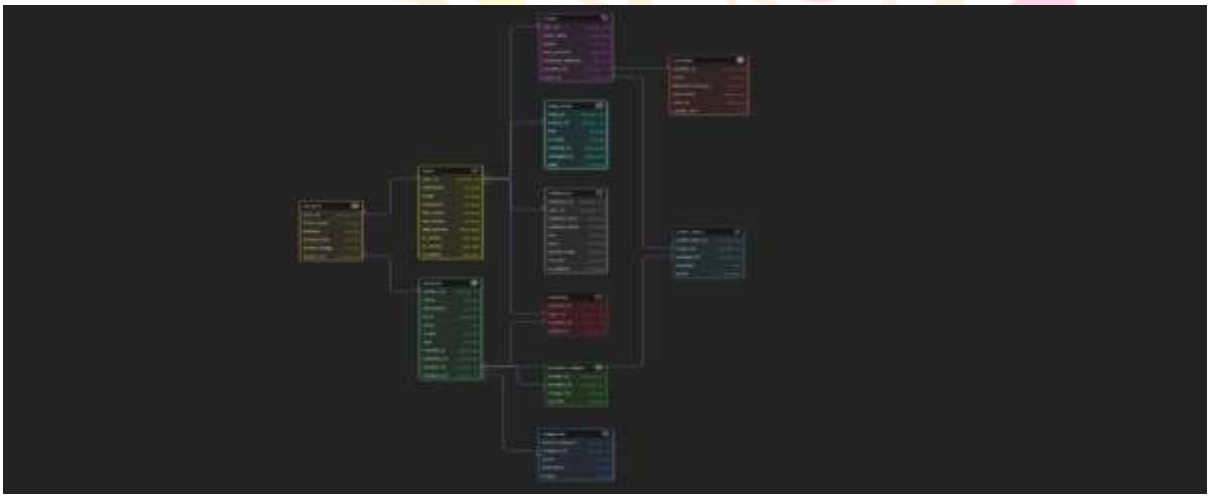

Fig. 7 Data Base ER Diagram

## 5. DISCUSSIONS

These limitations are common in initial versions of web applications and provide a roadmap for future enhancements. Addressing them would make the platform more robust, scalable, and suitable for a wider range of users and business scenarios. Prioritizing improvements in payment integration, analytics, real-time features, and security will be essential for production-grade deployment and long-term success.

## 6. CONCLUSION

The development and deployment of the Django-based e-commerce platform mark a significant achievement in building a robust, scalable, and user-friendly online shopping solution. Throughout the project, a wide range of essential features were implemented, including user authentication, product and vendor management, shopping cart and order processing, wishlist, vouchers, blog, and address management. The integration of the Jazzmin admin interface further enhanced the efficiency and professionalism of backend operations.

The project successfully met its primary objectives by delivering a platform that is not only functional but also secure and adaptable to future needs. The modular architecture and use of Django's best practices ensure that the system can be easily maintained and extended as business requirements evolve.

## 7. FUTURE SCOPE

Integrate secure, real-world payment gateways (such as Razorpay, PayPal, Stripe, or Paytm) to support credit/debit cards, UPI, net banking, and wallets. Implement payment status tracking, refunds, and transaction history for users and admins.

Add support for multiple payment options and currencies. Develop comprehensive dashboards for admins and vendors, showing sales trends, product performance, customer demographics, and inventory status

## 8. ACKNOWLEDGEMENTS

Mr. Rongala Rajesh [Assistant Professor] is an enthusiastic and committed faculty member in the Department of Computer Science. As an early-career academician, he has shown strong dedication to student development through active involvement in project guidance and technical mentoring. Despite being at the beginning of his professional journey, he has effectively guided students in executing academic projects with precision and conceptual clarity. His passion for teaching, coupled with a solid understanding of core computer science principles, positions him as a promising educator and mentor. Mr. Rajesh continues to contribute meaningfully to the academic environment through his proactive approach to learning and student engagement.

Vakada Govardhini Lakshmi is pursuing her final semister MCA in Sanketika Vidya Parishad Engineering College, accredited with A grade by NAAC, affiliated by Andhra University and approved by AICTE. With interest in Python Django V Govardhini has taken up her PG project on Online Electronics E-commerce Platform and published the paper in connection to the project under the guidance of R Rajesh, Assistant Professor, SVPEC.

## 9. REFERENCES WITH LINKS

1. **Django Project Documentation**
Django Software Foundation. (2025). *Django Documentation (v5.x)*. Available at: https://docs.djangoproject.com/
2. **Python Official Documentation**
Python Software Foundation. (2025). *Python 3.13 Documentation*. Available at: https://docs.python.org/3/
3. **Jazzmin Admin Theme**
Jazzmin Team. (2025). *Jazzmin: A drop-in theme for Django admin*. Available at: https://django-jazzmin.readthedocs.io/
4. **Bootstrap Framework**
Bootstrap Team. (2025). *Bootstrap 5 Documentation*. Available at: https://getbootstrap.com/
5. **W3Schools Online Web Tutorials**
W3Schools. (2025). *HTML, CSS, JavaScript, SQL, Python Tutorials*. Available at: https://www.w3schools.com/
6. **Mozilla Developer Network (MDN) Web Docs**
Mozilla. (2025). *HTML, CSS, and JavaScript Documentation*. Available at: https://developer.mozilla.org/
7. **Stack Overflow**
Stack Overflow Community. (2025). *Django, Python, and Web Development Q&A*. Available at: https://stackoverflow.com/
8. **GeeksforGeeks Django Tutorials**
GeeksforGeeks. (2025). *Django Tutorials and Examples*. Available at: https://www.geeksforgeeks.org/django-tutorial/
9. **Real Python**
Real Python. (2025). *Django and Python Tutorials*. Available at: https://realpython.com/

10. **Django Packages**
Django Packages. (2025). *Django reusable apps, sites, tools, and more*. Available at: https://djangopackages.org/

11. **CKEditor for Django**
CKEditor Team. (2025). *Django CKEditor Documentation*. Available at: https://django-ckeditor.readthedocs.io/

12. **ShortUUID for Django**
Tom Christie. (2025). *ShortUUID Documentation*. Available at: https://github.com/skorokithakis/shortuuid

13. **SQLite Documentation**
SQLite Consortium. (2025). *SQLite Documentation*. Available at: https://www.sqlite.org/docs.html

14. **GitHub**
GitHub, Inc. (2025). *Source code hosting and collaboration platform*. Available at: https://github.com/

15. **YouTube Django E-commerce Tutorials**
Various Authors. (2025). *Django E-commerce Project Tutorials*. Available at: https://www.youtube.com/

16. **Django REST Framework**
Encode OSS Ltd. (2025). *Django REST Framework Documentation*. Available at: https://www.django-rest-framework.org/

17. **Book: Two Scoops of Django 3.x**
Greenfeld, D., & Roy Greenfeld, A. (2020). *Two Scoops of Django 3.x: Best Practices for the Django Web Framework*. Two Scoops Press.

18. **Book: Django for Professionals**
Vincent, W. (2022). *Django for Professionals: Production websites with Python & Django*. Independently published.

19. **Book: Test-Driven Development with Python**
Percival, H. (2017). *Test-Driven Development with Python: Obey the Testing Goat: Using Django, Selenium, and JavaScript*. O'Reilly Media.

20. **Research Article: E-commerce Security**
Laudon, K. C., & Traver, C. G. (2021). *E-commerce: business, technology, society*. Pearson.

21. **Django Signals and Authentication**
Simple is Better Than Complex. (2025). *Django Signals - A Complete Guide*. Available at: https://simpleisbetterthancomplex.com/

22. **Django Deployment Checklist**
Django Software Foundation. (2025). *Deployment checklist*. Available at: https://docs.djangoproject.com/en/5.0/howto/deployment/checklist/

23. **Django ORM and Query Optimization**
Vitor Freitas. (2025). *Django ORM Optimization Tips*. Available at: https://simpleisbetterthancomplex.com/

24. **Django Allauth Documentation**
Raymond Penners. (2025). *Django Allauth Documentation*. Available at: https://django-allauth.readthedocs.io/

25. **Django Channels Documentation**
Django Channels Project. (2025). *Django Channels Documentation*. Available at: https://channels.readthedocs.io/