ISSN: 2350-0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 8, Issue 5 , May 2021

Development of an Online Shop with Python Web Framework (Django)

Busari O. A., Adebisi O. A., Adeaga I. I., Oni A. A.

Department of Computer Engineering, The Polytechnic, Ibadan, Oyo State, Nigeria

ABSTRACT: Online Shopping is a lifestyle e-commerce web application, which retails various products. This paper allows viewing various products available and also enables users to purchase desired products instantly using Braintree payment processor which is a sub-division of PayPal payment processor. The main objective of this research is to develop an ecommerce website and integrate a payment processor. It also provides an easy access to Administrators and Managers to view orders placed by users. This paper discussed how Django is used to develop an ecommerce website along side with other third-party software and a relational database; SQLite which was used for storing data. Before installing Django for the development of this project, a python interpreter was installed on the host operating system of the platform used for developing the project; an example is a computer or a cloud service provider. Some other third-party modules used in the development of this project are: Django-Rosetta which aids the ease translation of language through the browser, ngrok which temporarily exposes the local web server to the internet for access by users during testing and Redis which recommends products to users based on the last purchase statistics. Conclusively, the goal of translation of language from English to other indigenous language was successfully achieved. A payment processor was integrated to allow payment for orders and a recommendation engine was also built.

KEYWORDS: Python, Django, Libraries, Redis, Ngrok, Virtual environment, Payment Processor.

I. INTRODUCTION

With the evolution of internet and so many other computing devices, market places have been brought to the fingertips of customers without leaving their homes, offices, and other places which can deny the customers access to the market at that particular moment. One significant outcome of the internet is the birth of online shopping which is also known as e-commerce (Electronic Commerce).

Many organizations nowadays invest so much in e-commerce, some of the major giants are Amazon, Shopify, AliExpress, Jumia, Konga, etc. These major organizations major role is to serve as middlemen between the manufacturers and consumers also known as an online retailer between manufacturers and consumers. The importance of such service is so significant because consumers cannot always be at all the market places at the same time to purchase goods, for that reason, these major players in this business deal with goods from a variety of manufacturers from different industries ranging from edible goods (groceries, desserts, etc.) to non-edible goods (like computers and electronics, wears, utensils etc.) and so on. This process of goods supply chain still remains the same as the normal one and has relieved consumers the stress of going to different market places to purchase goods.

Since the stress of going to different market places by the consumers have been cared for, some consumers still encounter some difficulties when using the services of this major organizations which may be due to language barrier.

The paper focuses on removing the barrier thereby converting the official spoken language (English) to either a native language or any other language. The aim of this research is to develop an online shop with a popular python web framework called Django. The objectives are as follows: To explore the functionalities of the technologies driving some popular e-commerce sites; To develop an e-commerce website translated to some indigenous languages; To integrate a payment processor for managing payments for customers on the site; To build a recommendation engine to recommend goods related to the item being purchased at the moment.

This paper shall explore the functionalities of a standard e-commerce website and other third-party software used for the development of an e-commerce website which are all based on the popular python web framework (Django).

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 8, Issue 5 , May 2021

II. LITERATURE REVIEW

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aims to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library [1].

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. Django is a free and open-source web framework, which follows the model-view-template architectural pattern. Django's primary goal is to ease the creation of complex, database-driven websites. The framework emphasizes reusability and "pluggability" of components, less code, low coupling, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings files and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models. Some well-known sites that use Django include the Public Broadcasting Service, Instagram, Mozilla, The Washington Times, Disqus, Bitbucket, and Nextdoor. It was used on Pinterest, but later the site moved to a framework built over Flask [2].

WEBSITE ARCHITECTURE FOR ECOMMERCE

Having a great site architecture means making products and categories findable on your website, in a way that users and search engines can reach them as efficiently as possible. There are two concepts regarding site architecture: Efficient crawling and indexing. This refers to the Technical Architecture or TA and Classifying, labeling and organizing content. This refers to Information Architecture or IA. Together, information and technical architecture form the site architecture (SA). A good understanding of these two concepts will help to build search engine optimized websites that are search-engine and user-friendly [3].
Information Architecture: is the process of classifying and organizing content on a website while providing user-friendly access to that content, via navigation. This process is done (or should be done) by information architects.
Technical architecture: is the process of designing the technical and functional aspects of a site. This is mostly done by web developers [4].

III. METHODOLOGY

The requirements for the development are listed as follows:
1. Python 3.x
2. Django 2.x
3. Operating system (windows, Linux, mac)
4. Redis (for recommendation engine)
5. Ngrok (to allow you to expose a web server running on our local machine to the internet.)

A. Creating the Virtual Environment

After python has been installed on the pc, a virtual environment module needs to be installed on the pc through python's package manager. The virtual environment makes it easy to run different versions of Django in isolation without interrupting the process of one another. The Django module is installed in the virtual environment along with some other modules that are needed to develop the application specifying their versions.

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 8, Issue 5 , May 2021

B. Creating various Apps

When the platform is ready for development, various applications can be created in the project which are relevant to the main idea of the website. For this research, the apps created are as follows: coupons, shop, cart, orders, and payment. Each of these apps comes with some important files when generated by the same Django-admin command in the command line of the operating system which allow programmers build the website easily and some are created by the programmer because it is not created by default by the command.

C. Creating a Superuser (admin)

The Superuser is simply the admin of the site. The admin account needs to be created from the command line through the Django-admin command. The admin account must be created in order to manage the site with a higher privilege than the users of the site. The admin has the privilege to create, retrieve, update, and delete data content and users from the site through the admin site. All models present in the models.py file must be registered in admin.py file in order to allow the models to be visible for the admin.

D. Creating models

A model is the single, definitive source of information about your data. It contains the essential fields and behaviors of the data you're storing. Each attribute of the model represents a database field.

E. Creating views for the models

When the models have been successfully registered in the admin site, creating views for users is another task needed to be accomplished. This refers to the logical functionality between the request and response of the clients and servers. There are two types of views:

1. **Function-based views:** A view function, or view for short, is simply a Python function that takes a Web request and returns a Web response. This response can be the HTML contents of a Web page, or a redirect, or a 404 error, or an XML document, or an image or anything. The view itself contains whatever arbitrary logic is necessary to return that response. This code can live anywhere you want, as long as it's on your Python path. For the sake of putting the code somewhere, the convention is to put views in a file called views.py, placed in your project or application directory. The view function returns an HTML page that includes the request of the user.

2. **Class-Based views:** Class-based views provide an alternative way to implement views as Python objects instead of functions. These allow you to structure your views and reuse code by harnessing inheritance and mixins. They do not replace function-based views, but have certain differences and advantages when compared to function-based views:

i. Organization of code related to specific HTTP methods (GET, POST, etc.) can be addressed by separate methods instead of conditional branching.

ii. Object oriented techniques such as mixins (multiple inheritance) can be used to factor code into reusable components.

F. Integrating a Payment Processor

A payment gateway allows you to process payments online. Using a payment gateway, you can manage customer's orders and delegate payment processing to a reliable, secure third party. You won't have to worry about processing credit cards in your own system. There are several payment gateway providers to choose from.

G. How does Django work?

To truly appreciate Django, you will need to peek under the hood and see the various moving parts inside. This can be both enlightening and overwhelming.

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 8, Issue 5 , May 2021

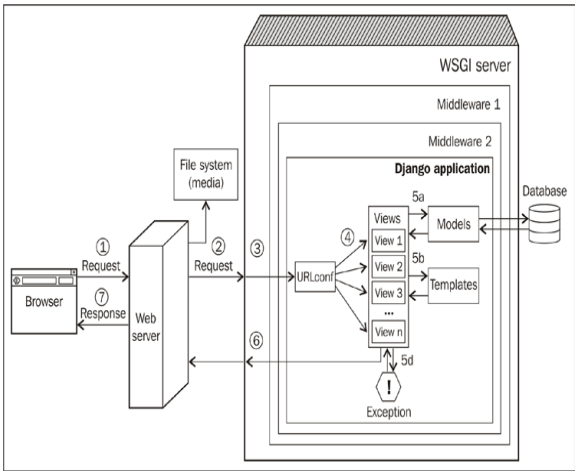

Figure 1: How web requests are processed in a typical Django application.

Figure 1 shows the simplified journey of a web request from a visitor's browser to your Django application and back. The numbered paths are as follows:

1. The browser sends the request (essentially, a string of bytes) to the web server.
2. The web server (say, Nginx) hands over the request to a WSGI server (say, uWSGI) or directly serves a file (say, a CSS file) from the filesystem.
3. Unlike a web server, WSGI servers can run Python applications. The request populates a Python dictionary called environ and, optionally, passes through several layers of middleware, ultimately reaching your Django application.
4. URLconf contained in the urls.py of your application selects a view to handle the request based on the requested URL. The request has turned into HttpRequest (a Python object).
5. The selected view typically does one or more of the following things:
    a. Talks to a database via the models
    b. Renders HTML or any other formatted response using templates
    c. Returns a plain text response (not shown)
    d. Raises an exception
6. The Http Response object gets rendered into a string, as it leaves the Django application.
7. A beautifully rendered web page is seen in your user's browser.

H. Building Recommendation Engine

A recommendation engine is a system that predicts the preference or rating that a user would give to an item. The system selects relevant items for the users based on their behavior and the knowledge it has about them. Nowadays, recommendation systems are used in many online services. They help users by selecting the stuff they might be interested in from the vast amount of available data that is irrelevant to them. Offering good recommendations enhances user engagement. E-commerce sites also benefit from offering relevant product recommendations by increasing their average sale.

Redis was used to store products that are purchased together. Redis (Remote Dictionary Server) is an in-memory data structure project implementing a distributed, in-memory key-value database with optional durability. Redis supports different kinds of abstract data structures, such as strings, lists, maps, sets, sorted sets, HyperLogLogs, bitmaps, streams, and spatial indexes.

IV. RESULT AND DISCUSSION

The online shop project built with python web framework (Django) has different applications which are as follows: cart, coupons, payment, shop and orders.

1. **Shop:** The catalog of our shop will consist of products that are organized into different categories. Each product will have a name, optional description, optional image, price, and availability. The Shop application has two entities in

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 8, Issue 5 , May 2021

its models.py file which are Category and Product. The Category serves as foreign key (a many-to-one relationship) to the Product because each product belongs to one category. Both entities are also registered in the admin.py file to make it shown in the admin panel of the website.

2. Cart: The Cart application is created for managing shopping carts. In the Cart application, a cart.py is created to manage sessions of user request. First, the user request tries to get the cart from the current session. If no cart is present in the session, an empty cart is created by setting an empty dictionary in the session. The cart dictionary uses product IDs as keys and a dictionary with quantity and price as the value for each key. By doing so, it can be guaranteed that a product is not added more than once in the cart; this way it is also easy to simplify the way to retrieve cart items.

3. Orders: The Orders application will contain information about customers and the products they are buying. When a shopping cart is checked out, the order needs to be saved into the database. The Orders application has two entities in its models.py file which are as follows: Order and OrderItem. The Order model contains several fields to store customer's information while the OrderItem model allows us to store the product, quantity, and price paid for each item. The two entities are also registered in the admin.py file to make it shown in the admin panel of the website. They are both under the order section in the admin panel.

4. Payment: The Payment application is created for managing payments for orders by customers that purchased goods on the website. In the Payment application after successfully creating an order, the order id in the current session is used as the session key in order to keep track of the goods purchased.

5. Coupons: The Coupon application is created to manage online coupon usually consisting of a code that is given to users, valid for a specific time frame. The coupon codes created will be valid for clients that enter the coupon in a specific time frame. The coupons will not have any limitations in terms of the number of times they can be redeemed, and they will be applied to the total value of the shopping cart.

The Online Shop (Images)

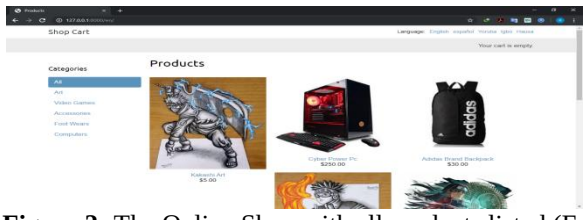

Figure 2: The Online Shop with all products listed (English)

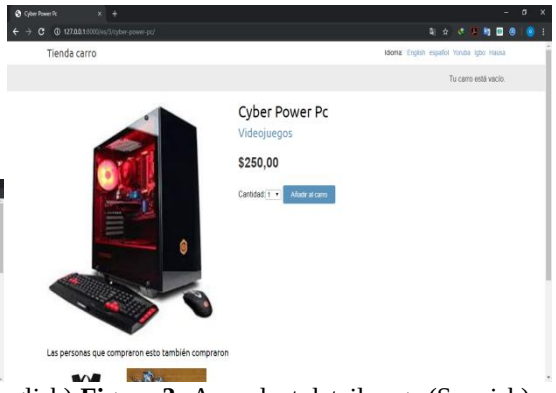

Figure 3: A product detail page (Spanish)

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 8, Issue 5 , May 2021

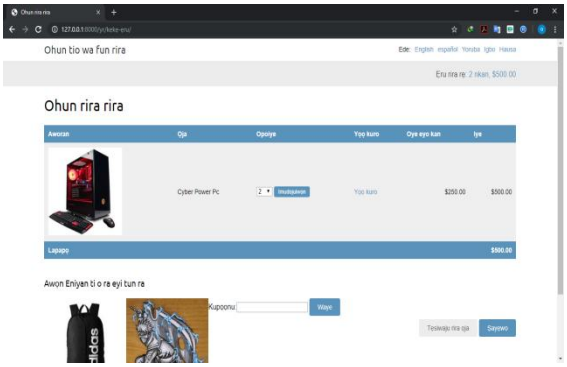

Figure 4: The Cart Detail Page (Yoruba)

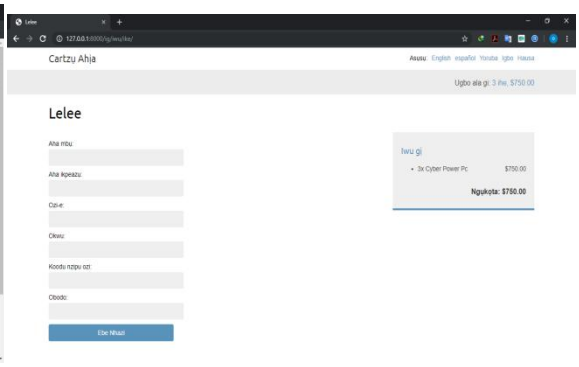

Figure 5: The Order Form (Igbo)

Figure 6: Payment Successful Page (Hausa)

The Admin Panel

The Admin panel or site is an interface where a user with administrator privileges (i.e. username and password) can access. The user of this interface is mostly the Administrator of the site. The Admin can create, retrieve, update and delete content on the website's database. In this project's Admin panel, there are few data models that are present.

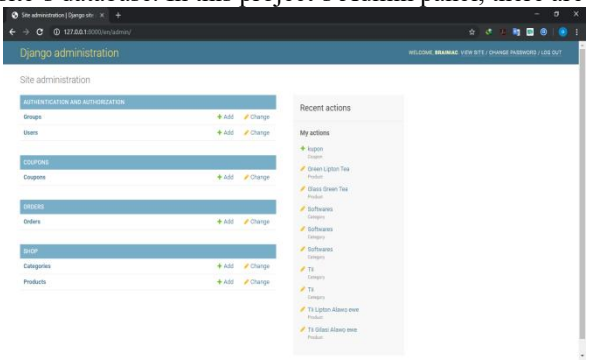

Figure 7: The Django Admin Panel

From Figure 7 above, the Groups and Users models under AUTHENTICATION and AUTHORIZATION are created automatically when the project was created, while the Coupons, Orders, Categories, and Products are created when each of the various applications created register its data model in the admin.py file. The data models shown in the Figure 7 can be Created, Retrieved, Updated and Deleted easily by the Admin using the Admin panel interface, the CRUD (Create, Retrieve, Update, and Delete) operations can also be performed using a shell. Some other functionalities that can be done are filter (filter by any choice possible i.e. date, name, category, etc.), extend, order_by, etc.

V. CONCLUSION AND RECOMMENDATION

In conclusion, the goal of translating the language from English to some indigenous languages was successfully accomplished. Additionally, a payment processor was also integrated to allow payment for orders by users and also a recommendation engine was built to suggest products to buy to customers based on past customers' purchasing statistics.

Base on the conclusion reached, the following recommendations were made:

ISSN: 2350-0328

International Journal of Advanced Research in Science, Engineering and Technology

Vol. 8, Issue 5 , May 2021

i. The need for security patches in the future so as to ensure a safer and an encrypted transfer of users' information online.

ii. The need for scalability to encompass future consumable products by consumers.

iii. The addition of more languages to the existing ones.

iv. The update of various third-party software used in the development of this project.

REFERENCES

[1] Python (Programming Language). (2019, May 10). In Wikipedia, The Free Encyclopedia. Retrieved 10:52, May 10, 2019, from https://en.wikipedia.org/wiki/Python_(programming_language).
[2] Django (Python Web Framework). (2019, May 10). In Wikipedia, The Free Encyclopedia. Retrieved 10:52, May 10, 2019, from https://en.wikipedia.org/wiki/Django_(web_framework).
[3] Website Architecture for Ecommerce. Retrieved 10:00, November 17, 2019, from https://ecommercetuners.com/ecommerce-site-architecture.
[4] Website Architecture for Ecommerce. Retrieved 10:00, November 17, 2019, from https://ecommercetuners.com/crawl-optimization-for-ecommerce.