# CHAPTER ONE INTRODUCTION

## **Background of Study**

Internet has contacted our lives from such a large number of points. One of the critical zones of day by day lives where it is profoundly influenced is matter of fact. Online business is where Information Technology (IT) and business exercises are merging into one another. It is a plan of action, or fragment of a bigger plan of action, that empowers a firm or individual to direct business over an electronic system, ordinarily the web (SAHA, 2018).

In this virtual time, a relevant online business procedure is the way to long haul sustenance for any business. Today, pretty much each and every errand is done through on the web and shopping is no exemption. Having a web-based business site is no more a way to accomplish upper hand. It is a vital basic for business, while customers are now flourishing on the web (SAHA, 2018).

Internet based business is quick making progress as an acknowledged and utilized business worldview. More entrepreneurs executing web-based business usefulness and online exchange framework into their sites. So it isn't difficult to state that into couple of decades web will be the typical for shopping any sort of item or administrations (SAHA, 2018).

The goal of this project is to expand a modern dynamic e-commerce internet software as online or digital store on the net. Where one of a kind varieties of product may be offered from the consolation of domestic through the internet.

## **Statement of the Problem**

As per an ongoing report by User Study and Experience Research Hub (Userhub) there is no single internet business webpage accessible in Nigeria that is completely consistent with universally settled web measures. The gathering of specialists who led the examination, which was centered around the commonness of Accessibility Errors, Accessibility Alerts, and Contrast Errors, and HTML and CSS approval. This test has been done on 174 dynamic internet business destinations recorded with the web-based business Association of Bangladesh (e-CAB). The specialists utilized W3C prescribed instruments and arbitrarily chosen 3 pages of every site for the testing reason on above issues.

The consequences of the examination demonstrated that, in Nigeria there were no single internet business locales with zero blunders and alerts. The normal commonness of openness blunders was 60.57, HTML mistakes 49.52, and CSS mistakes 27.16 (SAHA, 2018). This kind of related news inspired me to do the venture.

## **Aim and Objectives of the Study**

Is to develop a modern responsive ecommerce web application using react native & Django with strapi for customizable backend services.

The objectives are:

1. An engaging and easy to use UI with good UX using React Native.
2. The logic will be handled and the data will be stored using Django & Strapi.
3. Evaluating and verifying the app.

## Scope of the Project

This will be a multi shop ecommerce app allowing user to upload multiple products with their corresponding inventory and price.

## Limitation of the Study

This will be only on a local host as the web application is design for an undergraduate program.

## Significance of the Study

The platform or web app stands to bring a responsive and user experience interface design for user to facilitate easy use of the platform.

## Project Organization

The project is made up of three chapters. Chapter one summarizes the introductory study on ecommerce applications, including the background of study, statement of the problem, aim and objectives, significance of the study, scope of the study and limitation of the study. Chapter Two is the review of related literature. Chapter three is proposed methodology.

# CHAPTER TWO LITERATURE REVIEW



## **Introduction**

A look at what the programming language and database used the project are and a review of related literature.

## **Programming Languages Used**

### **React Native**

React Native is a JavaScript framework for writing real, natively rendering mobile applications for iOS and Android. It’s based on React, Facebook’s JavaScript library for building user interfaces, but instead of targeting the browser, it targets mobile platforms. In other words: web developers can now write mobile applications that look and feel truly “native,” all from the comfort of a JavaScript library that we already know and love. Plus, because most of the code you write can be shared between platforms, React Native makes it easy to simultaneously develop for both Android and iOS (Eisenman, 2015).

Similar to React for the Web, React Native applications are written using a mixture of JavaScript and XML-esque markup, known as JSX. Then, under the hood, the React Native “bridge” invokes the native rendering APIs in Objective-C (for iOS) or Java (for Android) (Eisenman, 2015). Thus, your application will render using real mobile UI components, not webviews, and will look and feel like any other mobile application. React Native also exposes JavaScript interfaces for platform APIs, so your React Native apps can access platform features like the phone camera, or the user’s location (Eisenman, 2015).

React Native currently supports both iOS and Android, and has the potential to expand to future platforms as well. In this book, we’ll cover both iOS and Android. The vast majority of the code we write will be cross-platform. And yes: you can really use React Native to build production-ready mobile applications! Some anecdota: [Facebook](http://bit.ly/1YipO7A), [Palantir](http://bit.ly/1PPEiZH), and [TaskRabbit](http://bit.ly/1PPEjNg) are already using it in production for user-facing applications (Eisenman, 2015).

#### **Advantages of React Native**

The fact that React Native actually renders using its host platform’s standard rendering APIs enables it to stand out from most existing methods of cross-platform application development, like Cordova or Ionic. Existing methods of writing mobile applications using combinations of JavaScript, HTML, and CSS typically render using webviews (Eisenman, 2015). While this approach can work, it also comes with drawbacks, especially around performance. Additionally, they do not usually have access to the host platform’s set of native UI elements. When these frameworks do try to mimic native UI elements, the results usually “feel” just a little off; reverse-engineering all the fine details of things like animations takes an enormous amount of effort, and they can quickly become out of date (Eisenman, 2015).

In contrast, React Native actually translates your markup to real, native UI elements, leveraging existing means of rendering views on whatever platform you are working with (Eisenman, 2015). Additionally, React works separately from the main UI thread, so your application can maintain high performance without sacrificing capability. The update cycle in React Native is the same as in React: when props or state change, React Native re-renders the views. The major difference between React Native and React in the browser is that React Native does this by leveraging the UI libraries of its host platform, rather than using HTML and CSS markup (Eisenman, 2015).

For developers accustomed to working on the Web with React, this means you can write mobile apps with the performance and look and feel of a native application, while using familiar tools. React Native also represents an improvement over normal mobile development in two other areas: the developer experience and cross-platform development potential (Eisenman, 2015).

### **Strapi**

Strapi is a headless CMS that is used to develop websites, mobile applications, eCommerce sites, and APIs. It allows you to create an API without knowing anything about the backend or databases. The system builds APIs based on content models automatically, making it easy to view data in the CMS with Strapi examples (Gadhavi, 2022).

* Strapi CMS is a free, open-source headless CMS that uses an API to link your frontend to Strapi's backend.
* It's a developer-friendly open-source and free-to-use service.
* Strapi is simple to learn and use, and you can get work done in minutes.
* Strapi may be used with React, [TezJS](https://tezjs.io/?utm_source=radix-blog&utm_medium=text&utm_campaign=what-is-strapi), Vue, Nuxt.js, Next.js, Angular, Svelte, Sapper, and Flutter.

#### **Why Use Strapi?**

A headless CMS strategy makes it simple to get new greenfield projects up and running. Developers chop out chunks and use APIs to connect everything (Gadhavi, 2022).

Designers and UX experts can now employ new tools without the constraints of the past. Designers and UX professionals are able to unleash their full creativity once free of constraints (Gadhavi, 2022).

The basic line is that using a headless CMS gives you the flexibility to use any future technology that makes sense. You'll know right now that adjustments can be made in the future without requiring you to rewrite the entire codebase (Gadhavi, 2022).

### **Django**

Django is a high-level Python web framework that enables rapid development of secure and maintainable websites. Built by experienced developers, Django takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It is free and open source, has a thriving and active community, great documentation, and many options for free and paid-for support (hamishwillee & ozgurturkiye, 2022).

Django helps you write software that is:

**Complete**

Django follows the "Batteries included" philosophy and provides almost everything developers might want to do "out of the box". Because everything you need is part of the one "product", it all works seamlessly together, follows consistent design principles, and has extensive and up-to-date documentation (hamishwillee & ozgurturkiye, 2022).

Versatile

Django can be (and has been) used to build almost any type of website — from content management systems and wikis, through to social networks and news sites. It can work with any client-side framework, and can deliver content in almost any format (including HTML, RSS feeds, JSON, and XML) (hamishwillee & ozgurturkiye, 2022).

Internally, while it provides choices for almost any functionality you might want (e.g. several popular databases, templating engines, etc.), it can also be extended to use other components if needed (hamishwillee & ozgurturkiye, 2022).

**Secure**

Django helps developers avoid many common security mistakes by providing a framework that has been engineered to "do the right things" to protect the website automatically. For example, Django provides a secure way to manage user accounts and passwords, avoiding common mistakes like putting session information in cookies where it is vulnerable (instead cookies just contain a key, and the actual data is stored in the database) or directly storing passwords rather than a password hash (hamishwillee & ozgurturkiye, 2022).

A password hash is a fixed-length value created by sending the password through a [cryptographic hash function](https://en.wikipedia.org/wiki/Cryptographic_hash_function). Django can check if an entered password is correct by running it through the hash function and comparing the output to the stored hash value. However due to the "one-way" nature of the function, even if a stored hash value is compromised it is hard for an attacker to work out the original password (hamishwillee & ozgurturkiye, 2022).

Django enables protection against many vulnerabilities by default, including SQL injection, cross-site scripting, cross-site request forgery and [clickjacking](https://developer.mozilla.org/en-US/docs/Glossary/Clickjacking) (hamishwillee & ozgurturkiye, 2022).

**Scalable**

Django uses a component-based "shared-nothing" architecture (each part of the architecture is independent of the others, and can hence be replaced or changed if needed). Having a clear separation between the different parts means that it can scale for increased traffic by adding hardware at any level: caching servers, database servers, or application servers. Some of the busiest sites have successfully scaled Django to meet their demands (e.g. Instagram and Disqus, to name just two) (hamishwillee & ozgurturkiye, 2022).

**Maintainable**

Django code is written using design principles and patterns that encourage the creation of maintainable and reusable code. In particular, it makes use of the Don't Repeat Yourself (DRY) principle so there is no unnecessary duplication, reducing the amount of code. Django also promotes the grouping of related functionality into reusable "applications" and, at a lower level, groups related code into modules (along the lines of the Model View Controller (MVC) pattern) (hamishwillee & ozgurturkiye, 2022).

**Portable**

Django is written in Python, which runs on many platforms. That means that you are not tied to any particular server platform, and can run your applications on many flavors of Linux, Windows, and macOS. Furthermore, Django is well-supported by many web hosting providers, who often provide specific infrastructure and documentation for hosting Django sites (hamishwillee & ozgurturkiye, 2022).

In a traditional data-driven website, a web application waits for HTTP requests from the web browser (or other client). When a request is received the application works out what is needed based on the URL and possibly information in POST data or GET data. Depending on what is required it may then read or write information from a database or perform other tasks required to satisfy the request. The application will then return a response to the web browser, often dynamically creating an HTML page for the browser to display by inserting the retrieved data into placeholders in an HTML template (hamishwillee & ozgurturkiye, 2022).

Django web applications typically group the code that handles each of these steps into separate files:

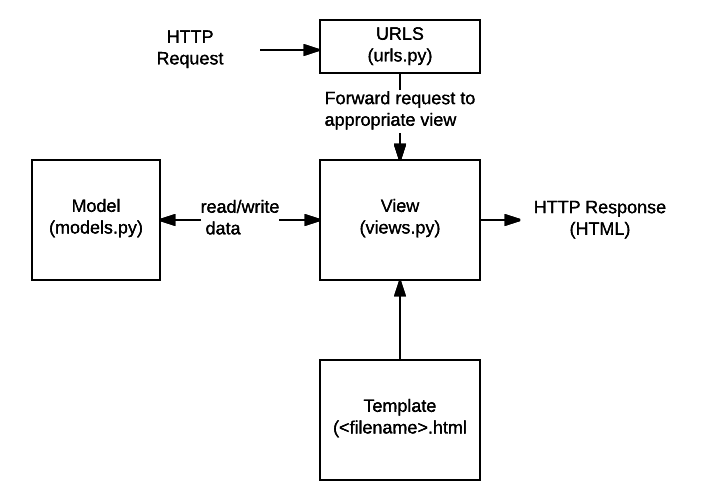


Figure 2.1: Structure of Django File (hamishwillee & ozgurturkiye, 2022)

1. **URLs:** While it is possible to process requests from every single URL via a single function, it is much more maintainable to write a separate view function to handle each resource. A URL mapper is used to redirect HTTP requests to the appropriate view based on the request URL. The URL mapper can also match particular patterns of strings or digits that appear in a URL and pass these to a view function as data (hamishwillee & ozgurturkiye, 2022).
2. **View:** A view is a request handler function, which receives HTTP requests and returns HTTP responses. Views access the data needed to satisfy requests via *models*, and delegate the formatting of the response to *templates* (hamishwillee & ozgurturkiye, 2022).
3. **Models:** Models are Python objects that define the structure of an application's data, and provide mechanisms to manage (add, modify, delete) and query records in the database (hamishwillee & ozgurturkiye, 2022).
4. **Templates:** A template is a text file defining the structure or layout of a file (such as an HTML page), with placeholders used to represent actual content. A *view* can dynamically create an HTML page using an HTML template, populating it with data from a *model*. A template can be used to define the structure of any type of file; it doesn't have to be HTML! (hamishwillee & ozgurturkiye, 2022).

## **Related Literature**

In this secion, literature from different sources (books, research paper and internet websites) on the ecommerce history, business model and comparative study has been discussed for better application development process. History of ecommerce says that web based shopping just wound up conceivable when the web was opened to general society in 1991. Amazon.com was one of the principal ecommerce destinations in the US to begin offering items on the web and a large number of organizations have pursued since. The tremendous fame of the web as of late has been powered generally by the possibility of performing business on the web. With the quick worldwide development in electronic trade, organizations are endeavoring to pick up an upper hand by utilizing internet business to interface with clients.

### **Definition of E-commerce**

There are numerous definitions for E-commerce which share a ton for all intents and purpose. By the by E-business has been characterized by researchers and specialists in various ways. Web based business alludes principally to the purchasing and offering exercises over the Internet, which incorporates exchanges, for example, setting orders, making installments, and following conveyance of requests on the Internet (Rodgers, 2002). The web based business is characterized as purchasing and offering of item administration or data through PC organizes mostly the web (Wen et al., 2001, as cited by SAHA, 2018) and individuals instantly consider shopper retail buys from organizations, for example, Amazon (Chaffey et al, 2006, as cited by SAHA, 2018). Be that as it may, internet business alludes to both monetary and educational electronically intervened exchanges between an association and any outsider it manages (Chaffey, 2006, as cited by SAHA, 2018). Further, Singh (2001, as cited by SAHA, 2018) upheld the definition by alluding web based business to an online administration offered to client to help their shopping knowledge over the web.

Online business is exponentially expanding the accessibility of data, giving clients access to more information, of preferable quality and quicker over previously. Mechanical headway drove an unstable improvement in electronic business, the reasons for that are the web end the World Wide Web (WWW), which are making electronic trade considerably more available. Web internet business incorporates electronic exchanging of physical merchandise and of intangibles, for example, data. This incorporates all the exchanging steps, for example, web based promoting, requesting, installment and support for conveyance (Trimmers, 2000, as cited by SAHA, 2018).

Online business is utilized wherever in regular daily existence. Presently a days it is used for everything from Visa approval, travel reservation over a system, wire support exchanges over the world, purpose of offer (POS) exchanges in retailing, electronic managing an account. It helps in create interest for the items and benefits and enhances arrange the executives, installment and other help capacities (Awad, 2004, as cited by SAHA, 2018). Thus, online business diminishes the expense of making, preparing, disseminating and recovering paper based data. Further encourages the advantages which incorporate enhanced picture, enhanced client benefit, streamlined procedures, packed cycle and conveyance time, disposing of printed material and expanded adaptability (Turban et al, 1999, as cited by SAHA, 2018).

Additionally, online business isn't just about purchasing and offering as it is likewise about electronically imparting, teaming up and finding data (Turban et al, 2004, as cited by SAHA, 2018). It can include the Internet, groupware programs, open email systems (Adams, 1994, as cited by SAHA, 2018), innovations, for example, electronic information trade (EDI) and electronic supports exchange (EFT) and all the more as of late, administrations related with cell phones and computerized TV (Voss, 1999, as cited by SAHA, 2018). E – Business has turned out to be exceptionally well known due to the advantages and accommodation it brings along as it is not any more an option, it is a goal (Wen et al, 2001, as cited by SAHA, 2018). E-trade is by and large dependent on financial exchange and e-administrations are the piece of electronic trade, as cited by (SAHA, 2018).

### **History of Ecommerce**

The historical backdrop of web based business begun 40 years prior and, right up 'til today, keeps on developing with new technological advancements, innovations, and a huge number of organizations entering the online market every year (SAHA, 2018).

Electronic Data Interchanges and mail order shopping during the 1970s made ready for the present day internet business store. The historical backdrop of online business is nearly entwined with the historical backdrop of the web. Web based shopping just ended up conceivable when the web was opened to the general population in 1991 (SAHA, 2018).

Web based shopping was designed and spearheaded in 1979 by Michael Aldrich in the United Kingdom. He associated an adjusted local TV by means of a phone line to an ongoing multi-client exchange handling PC (SAHA, 2018). The framework was promoted starting in 1980 and offered fundamentally business-to-business frameworks that were sold in the UK, Ireland, and Spain. One the most punctual purchaser shopping encounters was Book Stacks Unlimited, an online book shop made by Charles M. Stack in 1992 (SAHA, 2018). Stack's store started as a dial-up release board two years previously Amazon was established by Jeff Bezos. In 1994, Book Stacks Unlimited moved to the Internet as Books.com and was eventually acquired by Barnes & Noble (SAHA, 2018).

The main online exchange was, by a few reports, marijuana sold by Stanford students to MIT students by means of the Arpanet account at their artificial intelligence lab in 1972 (SAHA, 2018). In any case, the primary web based shopping exchange on the Internet occurred exactly 22 years after the fact. With the feature "The Internet is Open", the August 12, 1994, issue of New York Times chronicled the deal between two companions of a Sting CD. The Times said, “The team of young cyberspace entrepreneurs celebrated what was apparently the first retail transaction on the Internet using a readily available version of powerful data encryption software designed to guarantee privacy (SAHA, 2018).”

History of online business is unimaginable without Amazon and eBay which were among the principal Internet organizations to permit electronic exchanges. Because of their originators we presently have an attractive web based business area and appreciate the purchasing and offering points of interest of the Internet (SAHA, 2018).

Amazon is one of the primary web based business organizations to build up a partner promoting system, and these days the organization gets about 40% of its deals from associates and outsider venders who rundown and offer products on the site. In 2008 Amazon entered into the film and is right now supporting the film "The Stolen Child" with twentieth Century Fox (SAHA, 2018).

As per the exploration led in 2008, the space Amazon.com pulled in around 615 million clients consistently. The most well-known component of the site is the audit framework,

i.e. the capacity for guests to present their audits and rate any item on a rating scale from one to five stars. Amazon.com is additionally outstanding for its clear and user-friendly advanced search facility which empowers visitors to search for keywords in the full content of numerous books in the database (SAHA, 2018).

### **Different types of E-commerce**

The E-trade exchanges occur between opposite sides of an exchange. The sides of this exchange can be a customer, a business, inner or the administration. At the point when the transaction is between the business and the consumer, it is called B2C. The transaction between businesses is called B2B E-commerce (SAHA, 2018). For the same reason the when E-commerce happens between business and government, it is called B2G and C2C for consumer to consumer (SAHA, 2018).

The following table is a quick demonstration of different types of existed E-commerce/E- businesses (SAHA, 2018).

1. Not easy to classify
2. Many real models fall into multiple categories

TABLE 2.1: EXAMPLE OF DIFFERENT TYPE OF THE E-COMMERCE

|  |  |  |
| --- | --- | --- |
| **B2B**  Business to Business   1. PayPal (and B2C, B2B2C) 2. Optize (and B2C) 3. Alibaba Group | **B2C**  Business to consumer   1. Amazon 2. FreshDirect 3. Zynga (and C2C) | **B2G**  Business to Government   1. E-Procurement |
| **C2B**  Consumer to Business   1. ZonZoo 2. Fotolia 3. Google Adsense | **C2C**  Consumer to Consumer   1. Prosper (P2P) 2. eBay 3. Facebook | **C2G**  Citizen to Government   1. Agencia Tributaria (Tax agency online) |
| **G2B**  Government to Business   1. E-Government 2. AEPM 3. Certificado Digital | **G2C**  Government to Citizen   1. E-Government 2. eDNI 3. USA.gov (also G2B) | **G2G**  Government to Government   1. Government Gateway 2. Schengen Information System |

### **E-commerce Usage in Developed vs. Developing Countries**

Online business can possibly enhance proficiency and efficiency in numerous regions and, accordingly, has gotten critical consideration in numerous nations. Be that as it may, there has been some uncertainty about the significance of web based business for developing countries (SAHA, 2018). The nonappearance of satisfactory fundamental infrastructural, financial and the absence of government national ICT systems have made a huge hindrance in the adoption and development of online business in developing countries. For web based business in developing countries, social issues also should be considered (SAHA, 2018).

Wu et al. (2020) paper combs the concept of C2B according to the relevant literature research, through the comparative analysis of the decision relationship between B2C and C2B, discusses the essential connotation of C2B electronic commerce, and analyzes the C2B conceptual misunderstanding existing in the application of e-commerce combined with practical cases. On this basis, the C2B application platform and the main bottlenecks that may exist in the application of electronic commerce are discussed. Finally, the solution strategy of C2B application bottleneck and the future trend of C2B application are discussed.

E-commerce is mainly divided into three types, but (Xu & Chen, 2020) mainly analyzes the application of big data technology in B2C e-commerce precision marketing pattern. Firstly, the notion and features of B2C e-commerce as well as the concept and characteristics of precision marketing mode are described. Secondly, this paper further analyzes some challenges faced by B2C e-commerce in the times of large data. Finally, taking China Amazon B2C electronic commerce as an example, this paper briefly analyzes the ratio of B2C e-commerce in China’s online retail B2C market share in the third quarter of 2017, which is 4.1%. This shows that B2C e-commerce is developing rapidly under the background of large data era.

Qwaider (2020) introduces the concepts of e-commerce application framework based on cloud computing, the development trend of cloud computing that adapts to the problem of e-commerce and the storage and distribution of resources. A proposed structure allows organizations to reduce costs through the effective implementation of ecommerce activities and solve the problem of large companies to improve e-commerce applications through cloud computing.

Agrawal & Dhar (2021) reported in this paper primarily focuses on India’s top B2C organizations and the role of definitive user experience elements in shaping the success of these organizations. A heuristic-based comparative study was conducted among the top e-commerce Web applications in order to identify unique features that contribute to a sumptuous user experience. The analytical study was backed up by user studies that report the expectation of the customers and their frustrations. The collective insights were then correlated to each platform’s market ranking and share with an objective to identify and analyze features, design cues and elements that contribute to their standing in the Indian market. In a nutshell, the qualitative study explores connections between market ranking, usability heuristics, and user study insights in order to prescribe design features, cues, and elements that benchmark the user experience framework for B2C applications in an Indian context. The insights elaborate on the current trends, gaps, and opportunity areas for B2C applications.

Defiesta et al. (2020) discusses the development of an ecommerce platform with inventory integration for a mining equipment supplier companies. Because of e-commerce's low barrier of entry, many vendors and distributors sell directly to customers. With this, there is an opportunity to automate the business processes of the mid-size mining equipment supplier company to improve the service and sales performance. In turn, it can ease the management profit of the distributor channel. The system produces an e-commerce web application that can be overall access by the administrator. Likewise, distributors and customers' accounts can be partly access through the web application. This research includes modules to improve the system such as admin module, vendor module, distributor module, customer module and IT management system.

# CHAPTER THREE METHODOLOGY AND DESIGN



## **Introduction**

This chapter covers the analysis models (data flow diagram, sequence, class and entity relationship diagram) of the project and the higher-level solution (programming language) approach used.

## **Number of modules of e-commerce application**

This E-commerce project is divided into 9 modules:

* + Registration Module
  + Products Browse Module
  + Products Search Module
  + Shopping cart Module
  + Shipping & Billing Module
  + Payment Module
  + Admin User Management Module
  + Admin Catalog Management Module
  + Admin Order Management Module

## **Analysis Models**

Analysis models is a technical representation of the system. It acts as a link between the system description and the design model. In Analysis Modelling, information, behavior, and functions of the system are defined and translated into the architecture, component, and interface level design in the design modelling.

### **Data Flow Diagram**

A data flow diagram (DFD) maps out the flow of information for any process or system. It graphically represents the flow of data in a business information system.

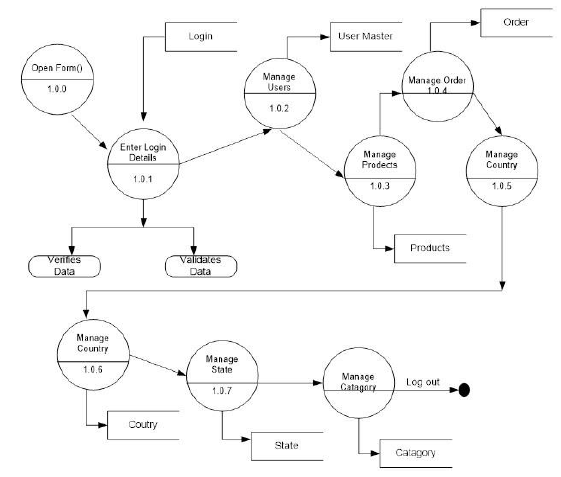


Figure 3.2: Data Flow Diagram

### **Sequence Diagram**

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration.

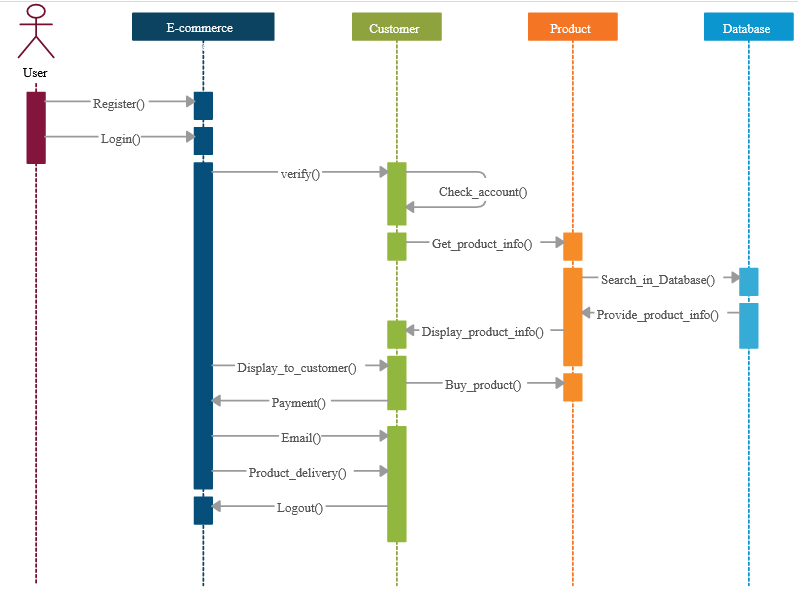


Figure 3.3: Sequence Diagram

### **Entity-Relationship Diagram**

An entity-relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties.

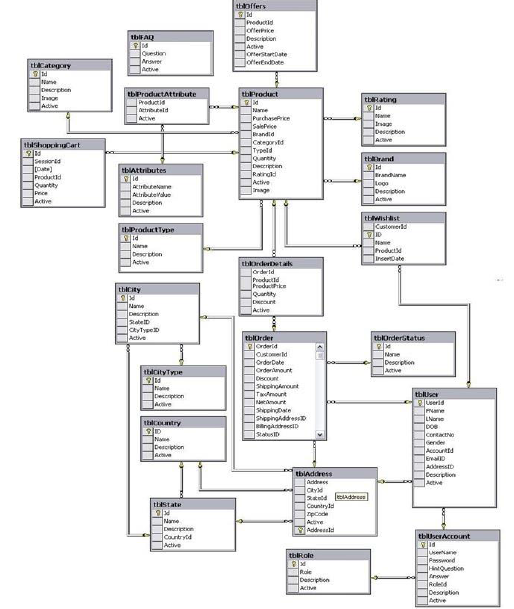


Figure 3.4: E.R Diagram

### **Architecture Diagram**

Class diagram is basically **a graphical representation of the static view of the system and represents different aspects of the application**. A collection of class diagrams represent the whole system. The name of the class diagram should be meaningful to describe the aspect of the system.



Figure 3.5: Class Diagram

## **Programming languages used**

1. React
2. Strapi
3. Python Django

**CHAPTER FOUR**

**SYSTEM IMPLEMENTATION EVALUATION**

**4.1 Introduction**

This section describes in detail how the new system will be implemented in order to assure its efficacy. It illustrates instances of functional (new) systems as well as how the system will be implemented.

* 1. **System Testing and Evaluation**

The developed system should be tested for a variety of reasons. For example, only via testing will we be able to detect and address any problems in the new system. Unit and integration testing were used in this project to confirm the design's efficacy and efficiency, as well as to ensure the new system satisfies its functional requirements and is error-free.

**Unit Testing**

specific units or single components of the system are examined individually in this part to confirm that specific phases function properly and without problems.

**Integration Testing**

The program was tested via integration testing, in which all of the components were integrated and worked as one. The connection between the different components was examined to ensure that they are correctly integrated and that the units can function as a unit.

**4.3 System Installation**

In order to use the proposed application on any computer system, the following steps need to be taken:

1. Make sure, pip, pipenv, and python3 or greater are installed on the system.
2. Copy your project folder to any location of your choice.
3. Open project folder in Visual Studio Code
4. On the terminal run “pipenv install -r requirements.txt”
5. On the terminal run “python manage.py runserver”
6. Open any browser on the system example Chrome, Microsoft Edge, or Mozilla Firefox.
7. On the address bar, type http://127.0.0.1 and press the enter key the site should be loaded.

**4.4 Security Measures**

Since the scope of the application is public, literally all the information is made available to any user, but some functionalities are restricted to the admin, functionalities that have to do with adding products, managing user accounts, etc are restricted from the general user. The restriction is carried out by using passwords when the application is accessed.

**4.5 Sample Outputs**

These describe and give the pictorial representation of the program or software; it shows and gives clear understanding of the design, and displays all the interfaces

**Homepage**

The image provided illustrates the homepage, which serves as the initial page and serves as a gateway to navigate and explore the various sections and functionalities of the website

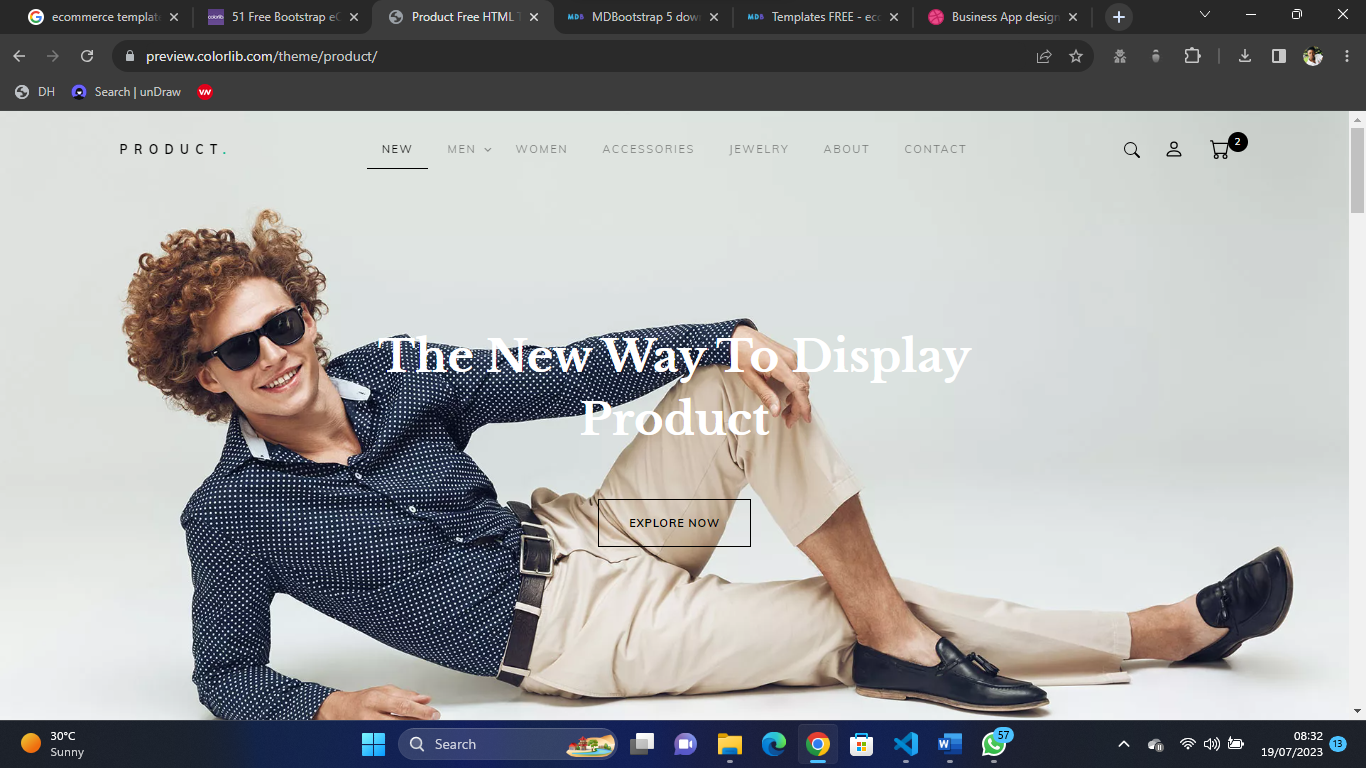


Fig 4.5.1 Homepage

**Products**

The page displays all the products available to the user for purchase

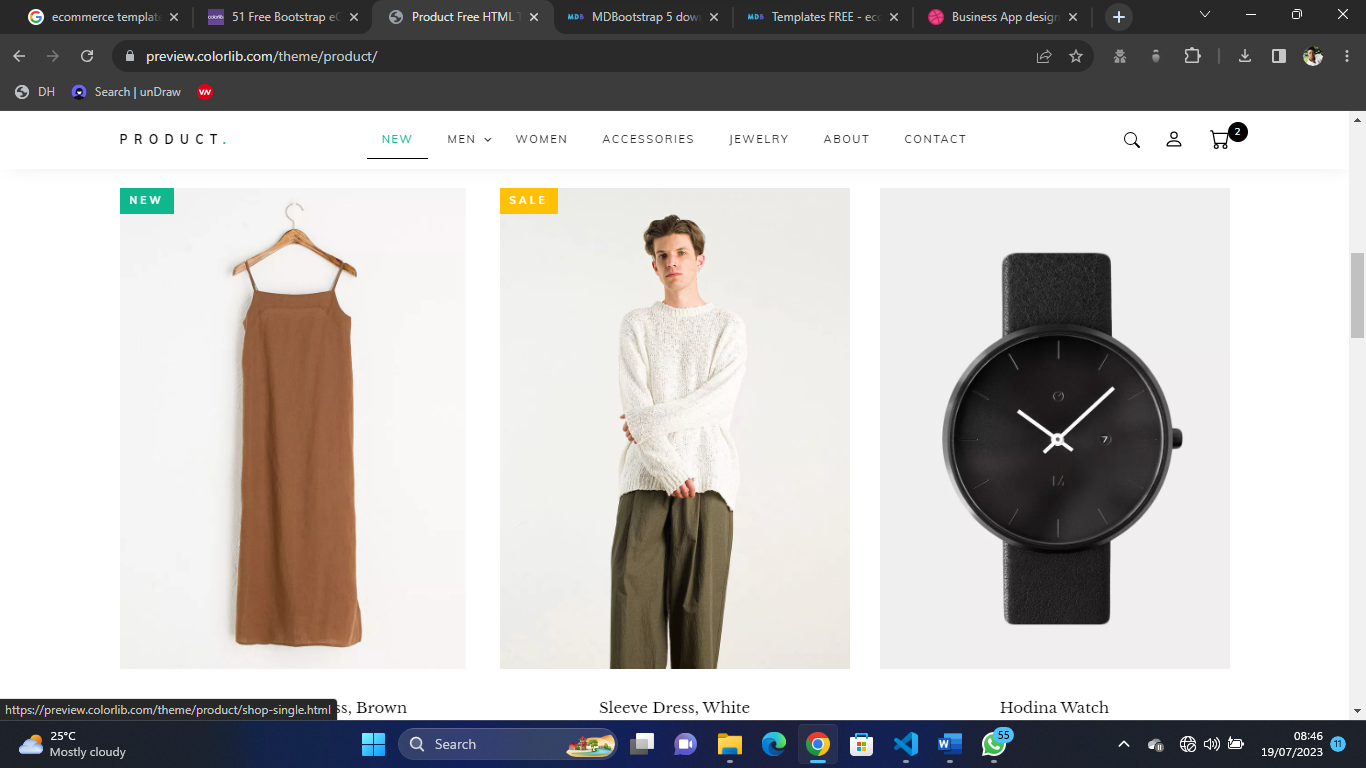


Fig 4.5.2 Products

**Product Categories**

This page highlights key categories of products that are available and users can click to purchase them.

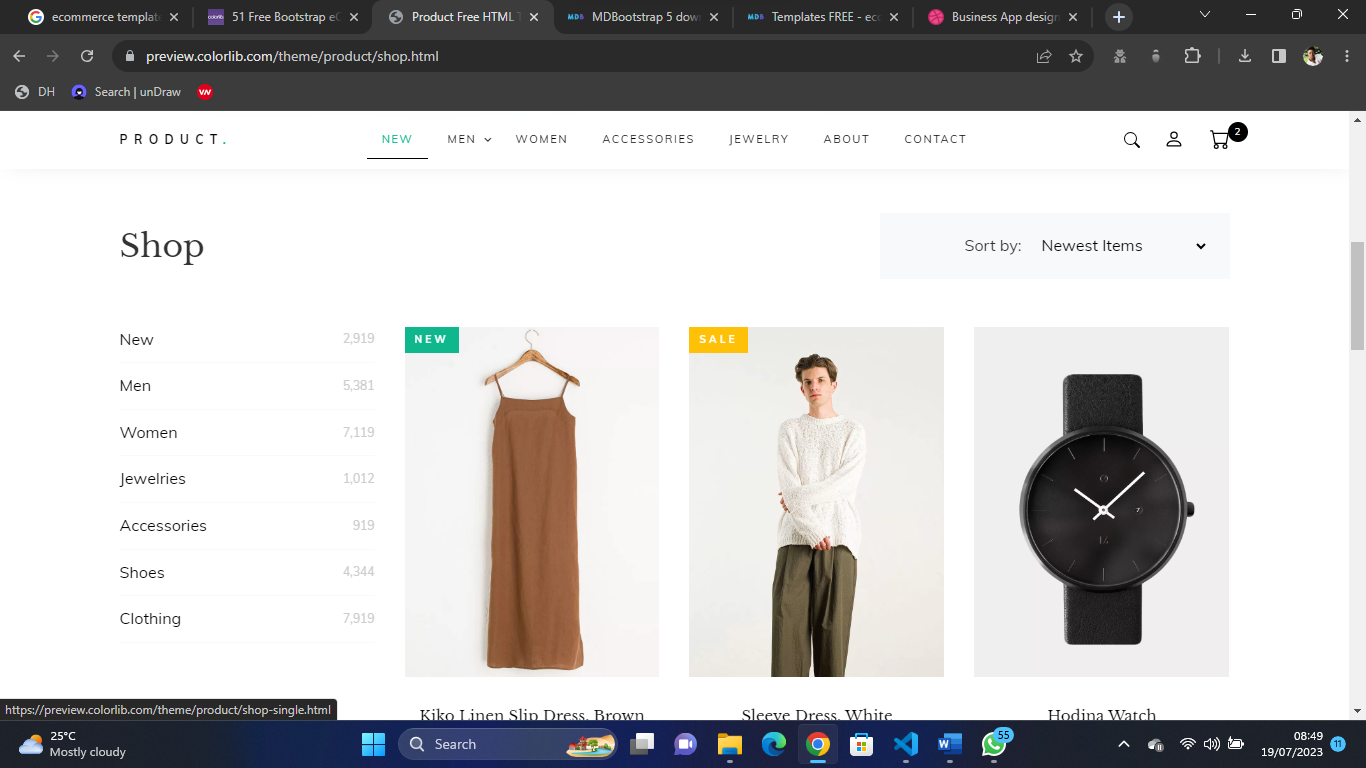


Fig 4.5.3 Product Categories

**Cart**

Authenticated users can view items in the cart, and increase or decrease the items before proceeding to checkout

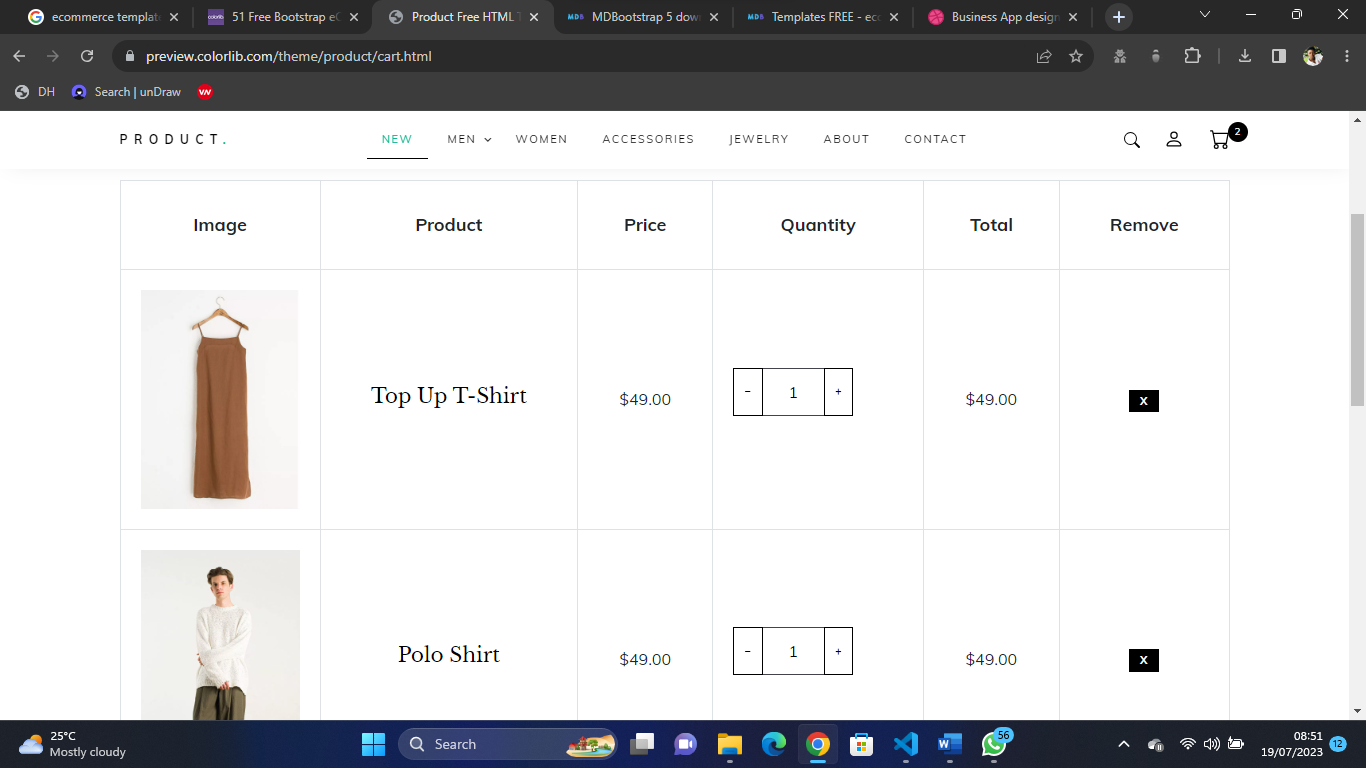


Fig 4.5.4 Cart

**About Us**

This page contains detailed reason why end-users should make use of our solution

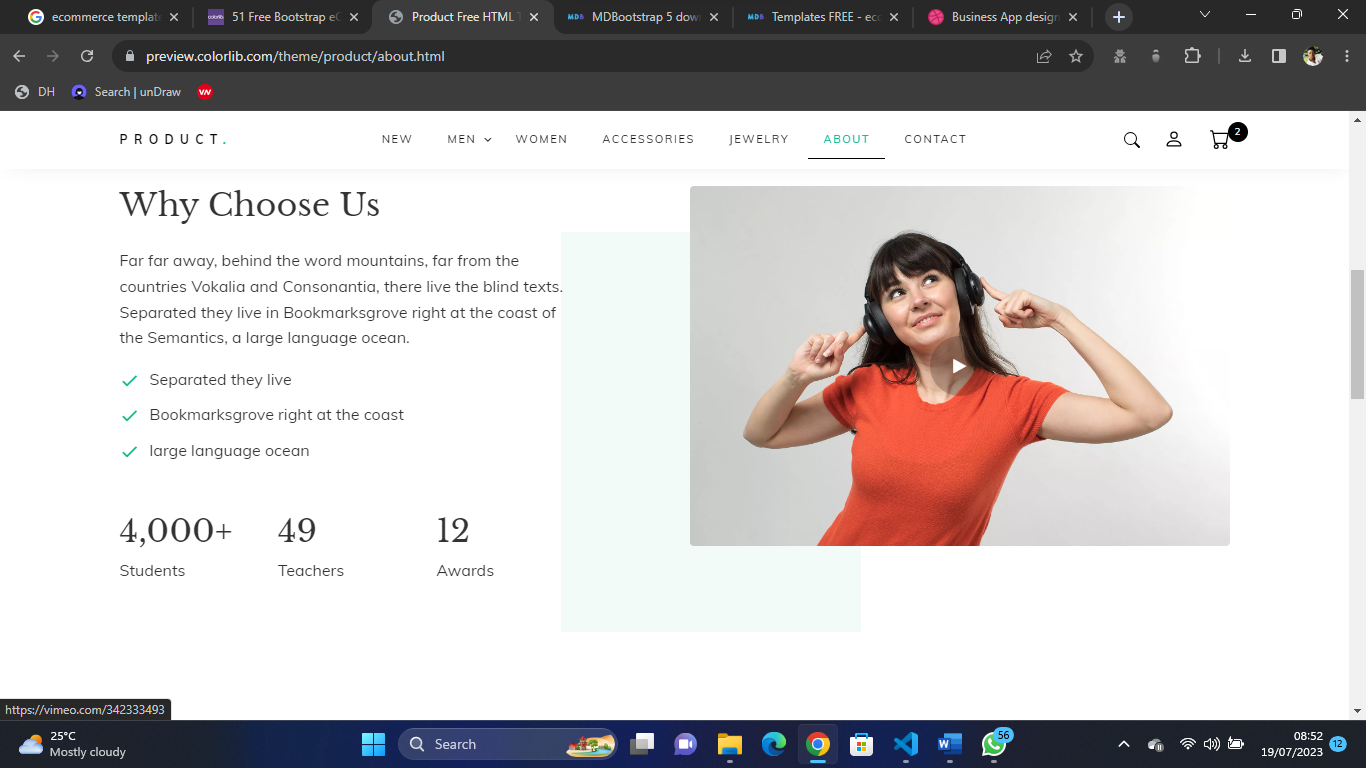


Fig 4.5.4 About Us

**Contact Us**

This is the screen where the user can contact the admin possibly because of any difficulty faced in using the application

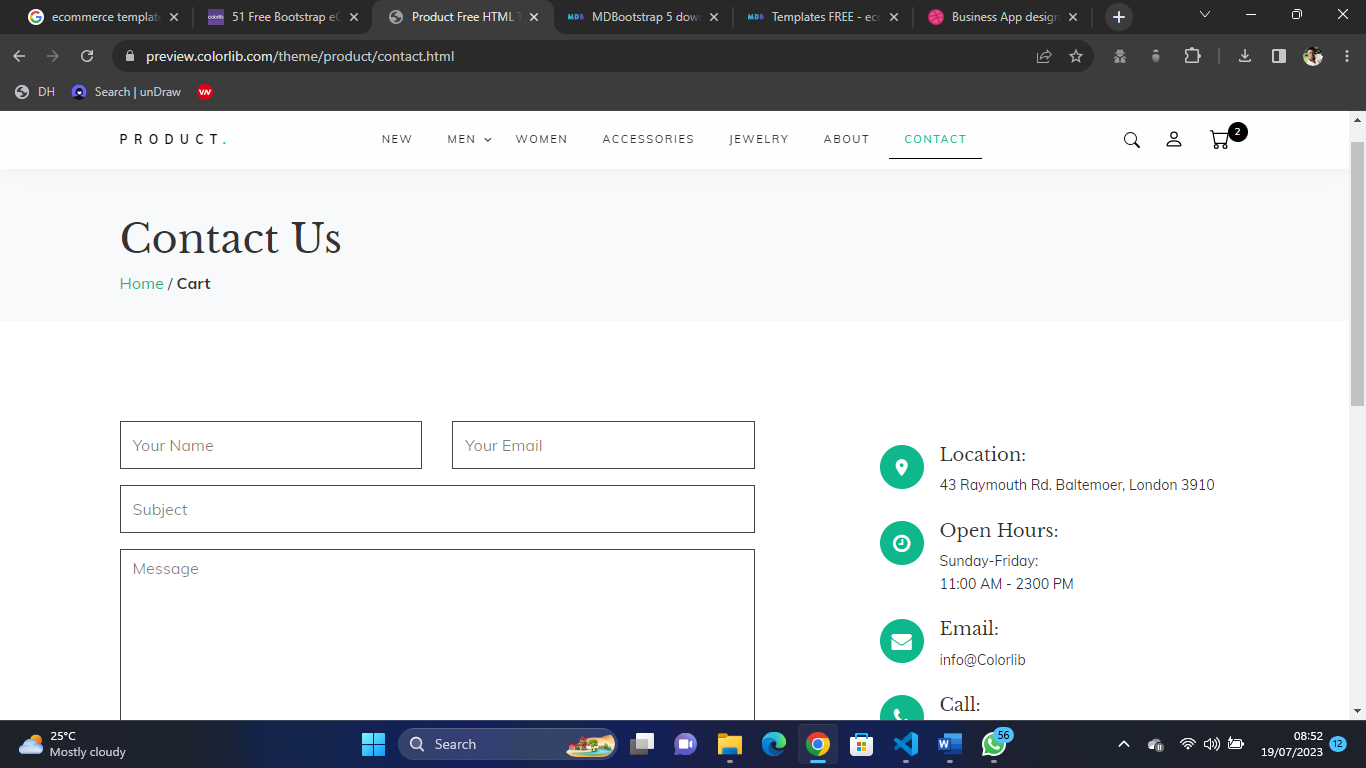


Fig 4.5.5 Contact Us

**Checkout Page**

This is the page where the user can checkout the placed order

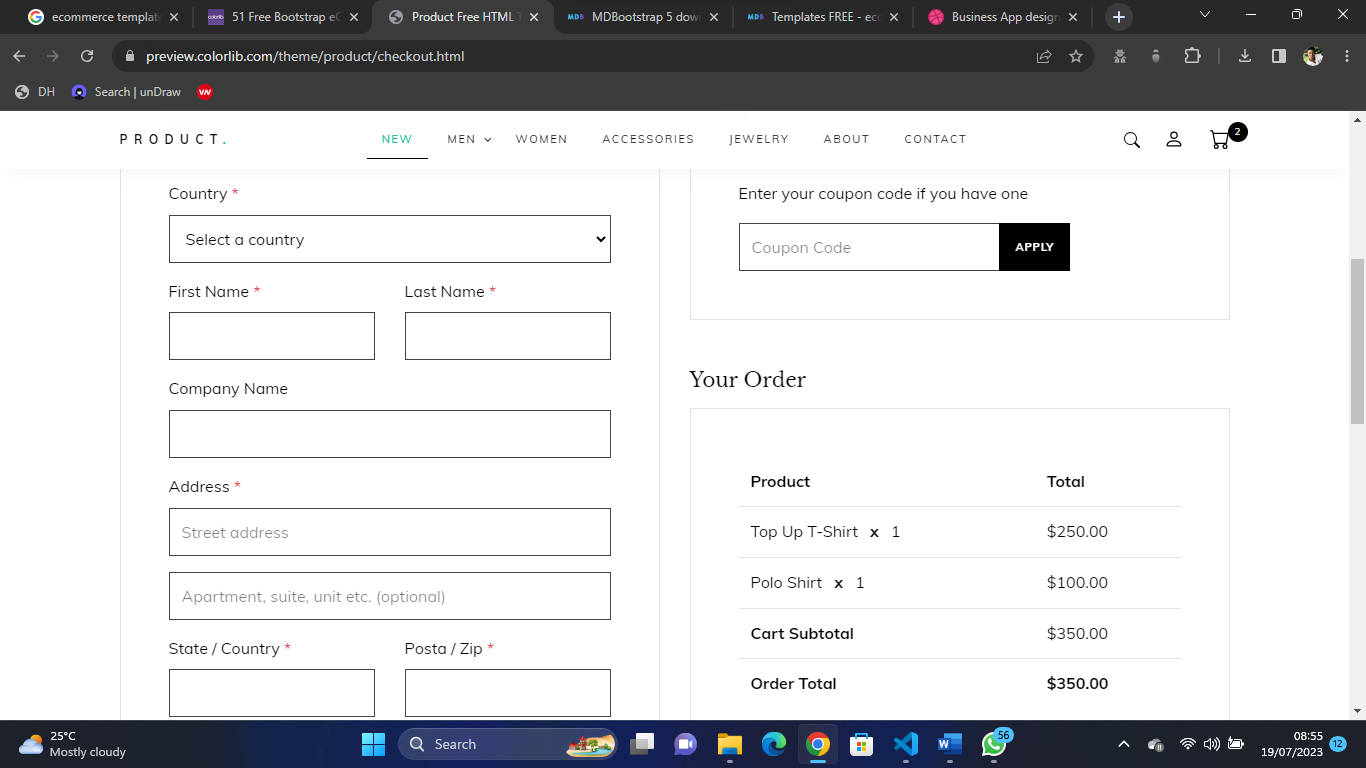


Fig 4.5.6 Checkout Page

**CHAPTER FIVE**

**SUMMARY CONCLUSION AND RECOMMENDATION**

**5.1 Summary**

The study aims to develop a modern responsive e-commerce web application using React Native and Django with Strapi for customizable backend services. The project's main objectives are to create an engaging and user-friendly UI with good UX using React Native and handle logic and data storage through Django and Strapi. The scope of the app will encompass multiple shops, allowing users to upload various products with inventory and prices. However, the study has limitations, as it will be hosted locally for an undergraduate program. Nevertheless, the significance of this project lies in providing a platform with a responsive and user-oriented interface, facilitating easy usage and enhancing the overall user experience for online shopping.

**5.2 Conclusion**

In conclusion, this project aims to address the growing importance of e-commerce in modern business by developing a dynamic and user-friendly web application. The study focuses on utilizing the power of React Native and Django with Strapi to create an engaging and responsive platform for multiple shops. By providing a customizable backend and intuitive user interface, the application seeks to enhance the overall user experience, making online shopping more accessible and convenient for customers. While the study is limited to a local host for an undergraduate program, its significance lies in the potential to revolutionize the way businesses conduct online transactions. By embracing the ever-evolving landscape of e-commerce, this project contributes to the continued growth of digital commerce and reinforces the importance of implementing internationally established web standards for enhanced accessibility and usability.

**5.2 Recommendation**

Based on the findings of the study, the following recommendations are proposed to improve the e-commerce web application and optimize its performance:

* 1. Continuous Testing and Improvement: Regularly test the application for accessibility errors, HTML, and CSS validation using W3C-recommended tools. Address and fix any issues promptly to ensure the website meets international web standards and enhances user experience.
  2. Security Measures: Implement robust security measures to protect user data, transactions, and the overall application from potential cyber threats. Regularly update security protocols to stay ahead of evolving security challenges.
  3. Scalability and Performance: Ensure the application is designed to handle increased user traffic and transactions. Optimize the backend services and database to ensure smooth performance even during peak times.
  4. Localization and Multi-language Support: Consider incorporating multi-language support and localization features to cater to a wider audience and expand the application's reach to different regions.

By implementing these recommendations, the e-commerce web application can evolve into a highly efficient, user-friendly, and successful platform, offering an enhanced online shopping experience to its users.

**REFERENCES**

Hamishwillee, K & Ozgurturkiye, R. (2022). *Django Introduction - Learn Web Development: MDN*. Learn web development | MDN. Retrieved December 13, 2022, from https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction

Saha, S. (2018). *Implementation Of An E-Shopping Website* (thesis). Daffodil International University, Daffodil Smart City.

Eisenman, B. (2015). *Learning react native*. O'Reilly Media.

Gadhavi, M. (2022, August 25). *What is Strapi? Strapi examples and why you should use it?* Radixweb. Retrieved December 24, 2022, from https://radixweb.com/blog/what-is-strapi

Wu, Q., Ma, J., & Wu, Z. (2020). Consumer-driven e-commerce: A study on C2B applications. *2020 International Conference on E-Commerce and Internet Technology (ECIT)*. https://doi.org/10.1109/ecit50008.2020.00019

Xu, G., & Chen, Q. (2020). On the application of large data technology in B2C E-Commerce Precision Marketing Mode. *Advances in Intelligent Systems and Computing*, 1197–1203. https://doi.org/10.1007/978-981-15-2568-1\_166

Qwaider, W. Q. (2020). A Conceptual Model for E-Commerce Applications based a Cloud Computing. *International Journal of Computer Applications (0975 – 8887)* , *176*(13), 18–22.

Agrawal, M., & Dhar, D. (2021). Enhancing user experience of e-commerce platforms—a case study of B2C applications in the Indian market. *Design for Tomorrow—Volume 1*, *1*, 503–516. https://doi.org/10.1007/978-981-16-0041-8\_42

Defiesta, M. L., Estrada, D. G., Veloso, A. N., Llesol, M. G., & Intal, G. L. (2020). B2B and B2C e-commerce platform for mining equipment suppliers. *Proceedings of the 2020 2nd International Conference on Management Science and Industrial Engineering*. https://doi.org/10.1145/3396743.3396772

**APPENDIX**

**Homepage**

{% extends 'base.html' %}

{% load static %}

{% block title %} Home{% endblock %}

{% block head %}

   {% include 'partials/head.html' %}

{% endblock %}

{% block body %}

<div class="hero\_area">

   <!-- header section strats -->

   {% include "partials/nav.html" %}

   <!-- end header section -->

   <!-- slider section -->

   <section class="slider\_section ">

      <div class="slider\_bg\_box">

         <img src="{% static 'images/slider-bg.jpg' %}" alt="">

      </div>

      <div id="customCarousel1" class="carousel slide" data-ride="carousel">

         <div class="carousel-inner">

            <div class="carousel-item active">

               <div class="container ">

                  <div class="row">

                     <div class="col-md-7 col-lg-6 ">

                        <div class="detail-box">

                           <h1>

                              <span>

                              Best Service

                              </span>

                              <br>

                              On Everything

                           </h1>

                           <p>

                             Delfak Nigeria Limited the sure construction material shop.

                           </p>

                           <div class="btn-box">

                              <a href="{% url 'auth:login' %}" class="btn1">

                              Get Started

                              </a>

                           </div>

                        </div>

                     </div>

                  </div>

               </div>

            </div>

            <div class="carousel-item ">

               <div class="container ">

                  <div class="row">

                     <div class="col-md-7 col-lg-6 ">

                        <div class="detail-box">

                           <h1>

                              <span>

                              Best Quality

                              </span>

                              <br>

                              On Everything

                           </h1>

                           <p>

                              Delfak Nigeria Limited the sure construction material shop.

                           </p>

                           <div class="btn-box">

                              <a href="{% url 'auth:login' %}" class="btn1">

                              Get Started

                              </a>

                           </div>

                        </div>

                     </div>

                  </div>

               </div>

            </div>

         </div>

         <div class="container">

            <ol class="carousel-indicators">

               <li data-target="#customCarousel1" data-slide-to="0" class="active"></li>

               <li data-target="#customCarousel1" data-slide-to="1"></li>

            </ol>

         </div>

      </div>

   </section>

   <!-- end slider section -->

</div>

<!-- footer start -->

{% include "partials/footer.html" %}

<!-- footer end -->

{% include "partials/script.html" %}

{% endblock %}

**Checkout page**

{% extends 'base.html' %}

{% load static %}

{% block title %} Checkout{% endblock %}

{% block head %}

{% include "partials/auth\_head.html" %}

{% include "partials/stripe.html" %}

{% endblock%}

{% block body %}

{% include "partials/preloader.html" %}

<div id="main-wrapper">

    {% include "partials/login\_nav.html" %}

    {% include "partials/sidebar.html" %}

    <div class="content-body">

        <div class="row page-titles mx-0">

            <div class="col p-md-0">

                <ol class="breadcrumb">

                    <li class="breadcrumb-item"><a href="{% url 'auth:dashboard' %}">Dashboard</a></li>

                    <li class="breadcrumb-item active"><a href="{% url 'auth:profile' user.id %}">Order Summary</a></li>

                </ol>

            </div>

        </div>

        <div class="container-fluid">

            {% include "partials/messages.html" %}

            <div class="row">

                <div class="col-lg-8 col-xl-8">

                    <div class="card">

                        <div class="card-body">

                            <h3 class="card-header mb-0">Checkout Form</h3>

                            <form class="card-body" method="post">

                                {% csrf\_token %}

                                <div class="row">

                                    <label for="admin\_card" class="col-12 col-form-label">Customer Details</label>

                                    <hr>

                                    <div class="col-md-6 col-sm-12">

                                        <label for="fullname" class=" col-form-label">Full Name</label>

                                        {{ form.fullname }}

                                        {% if form.fullname.errors %}

                                        <div class="alert alert-danger mt-2" role="alert">

                                        {{ form.fullname.errors }}

                                        </div>

                                        {% endif %}

                                        <div class="invalid-feedback">Please, enter your Full name</div>

                                    </div>

                                    <div class="col-md-6 col-sm-12">

                                        <label for="email" class=" col-form-label">Email</label>

                                        {{ form.email }}

                                        {% if form.email.errors %}

                                        <div class="alert alert-danger mt-2" role="alert">

                                        {{ form.email.errors }}

                                        </div>

                                        {% endif %}

                                        <div class="invalid-feedback">Please, enter your email</div>

                                    </div>

                                    <div class="col-md-6 col-sm-12">

                                        <label for="phone" class=" col-form-label">Phone Number</label>

                                        {{ form.phone }}

                                        {% if form.phone.errors %}

                                        <div class="alert alert-danger mt-2" role="alert">

                                        {{ form.phone.errors }}

                                        </div>

                                        {% endif %}

                                        <div class="invalid-feedback">Please, enter your Phone Number</div>

                                    </div>

                                    <div class="col-md-6 col-sm-12">

                                        <label for="address" class=" col-form-label">Address</label>

                                        {{ form.address }}

                                        {% if form.address.errors %}

                                        <div class="alert alert-danger mt-2" role="alert">

                                        {{ form.address.errors }}

                                        </div>

                                        {% endif %}

                                        <div class="invalid-feedback">Please, enter your address</div>

                                    </div>

                                </div>

                                <hr>

                                <button id="btnsubmit" class="btn btn-primary btn-lg btn-block waves-effect waves-light">Complete Sales</button>

                              </form>

                        </div>

                    </div>

                </div>

                <div class="col-lg-4 col-xl-4">

                    <div class="card">

                        <div class="card-header d-flex align-items-center justify-content-between">

                            <h5 class="mb-0">Your Cart</h5>

                            <small class="text-muted float-end">cart</small>

                        </div>

                        <div class="card-body">

                            <div class="basic-list-group">

                                <ul class="list-group">

                                    {% for order\_item in order.product.all  %}

                                    <li class="list-group-item d-flex justify-content-between align-items-center">{{ order\_item.product.title }} <span class="badge badge-primary badge-pill">{{ order\_item.quantity }}</span>

                                    </li>

                                    {% endfor %}

                                    <li class="list-group-item d-flex justify-content-between align-items-center"><b>Total</b> <span class="badge badge-primary badge-pill">₦{{ orders.get\_total }}</span>

                                </ul>

                            </div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

    {% include "partials/login\_footer.html" %}

</div>

<script>

    // Custom styling can be passed to options when creating an Element.

    var style = {

      base: {

        // Add your base input styles here. For example:

        fontSize: '16px',

        color: '#32325d',

      },

    };

    // Create an instance of the card Element.

    var card = elements.create('card', {style: style});

    // Add an instance of the card Element into the `card-element` <div>.

    card.mount('#card-element');

    card.on('change', function(event) {

      var displayError = document.getElementById('card-errors');

      if (event.error) {

        displayError.textContent = event.error.message;

      } else {

        displayError.textContent = '';

      }

    });

    var form = document.getElementById('payment-form');

    // Create a token or display an error when the form is submitted.

    var form = document.getElementById('payment-form');

    form.addEventListener('submit', function(event) {

      event.preventDefault();

      stripe.createToken(card).then(function(result) {

        if (result.error) {

          // Inform the customer that there was an error.

          var errorElement = document.getElementById('card-errors');

          errorElement.textContent = result.error.message;

        } else {

          // Send the token to your server.

          stripeTokenHandler(result.token);

        }

      });

    });

    function stripeTokenHandler(token) {

    // Insert the token ID into the form so it gets submitted to the server

    var form = document.getElementById('payment-form');

    var hiddenInput = document.createElement('input');

    hiddenInput.setAttribute('type', 'hidden');

    hiddenInput.setAttribute('name', 'stripeToken');

    hiddenInput.setAttribute('value', token.id);

    form.appendChild(hiddenInput);

    // Submit the form

    form.submit();

  }

  </script>

{% include "partials/auth\_script.html" %}

{% endblock %}

Order Summary

{% extends 'base.html' %}

{% load static %}

{% block title %} Order Summary{% endblock %}

{% block head %} {% include "partials/auth\_head.html" %} {% endblock%}

{% block body %}

{% include "partials/preloader.html" %}

<div id="main-wrapper">

    {% include "partials/login\_nav.html" %}

    {% include "partials/sidebar.html" %}

    <div class="content-body">

        <div class="row page-titles mx-0">

            <div class="col p-md-0">

                <ol class="breadcrumb">

                    <li class="breadcrumb-item"><a href="{% url 'auth:dashboard' %}">Dashboard</a></li>

                    <li class="breadcrumb-item active"><a href="{% url 'auth:order\_summary' %}">Sales Summary</a></li>

                </ol>

            </div>

        </div>

        <div class="container-fluid">

            {% include "partials/messages.html" %}

            <div class="row">

                <div class="col-lg-8 col-xl-8">

                    <div class="card">

                        <div class="card-body">

                            <h3 class="card-header mb-0">Sales summary</h3>

                            {% include "partials/summary.html" %}

                        </div>

                    </div>

                </div>

                <div class="col-lg-4 col-xl-4">

                    <div class="card">

                        <div class="card-header d-flex align-items-center justify-content-between">

                            <h5 class="mb-0">Summary</h5>

                            <small class="text-muted float-end">cart</small>

                          </div>

                        <div class="card-body">

                            <div class="basic-list-group">

                                <ul class="list-group">

                                    {% for order\_item in order.product.all  %}

                                    <li class="list-group-item d-flex justify-content-between align-items-center">{{ order\_item.product.title }} <span class="badge badge-primary badge-pill">{{ order\_item.quantity }}</span>

                                    </li>

                                    {% endfor %}

                                </ul>

                            </div>

                        </div>

                    </div>

                </div>

            </div>

        </div>

    </div>

    {% include "partials/login\_footer.html" %}

</div>

{% include "partials/auth\_script.html" %}

{% endblock %}