**CHAPTER ONE**

**INTRODUCTION**

**1.1 Introduction**

This chapter will describe the context in which this investigation was launched, the description of difficulties that led to this study, and the Aim and Objectives of the study as a prologue to subsequent aspects of this study. Others include the significance of the research, the scope of the work, study limitations, and definitions of technical terminologies.

**1.2** **Background of the Study**

Food and grocery shopping have undergone a revolution in every retail sector, with noticeable changes in customer purchasing behavior driven by high-income growth, changing lifestyles, and cost-effective and efficient online and mobile technology. Consumers' need for ease has grown as they devote less time to shopping and more to other activities, and their attention has frequently been diverted to virtual shopping as an alternative medium. Thus, the rapid expansion of the internet is altering the way people browse for and purchase items and services, and it has quickly become a worldwide phenomenon. As a result, one of the primary motivators behind customer preferences for online purchasing has been convenience. Although still in its early stages, online grocery is gaining pace and becoming increasingly incorporated into the everyday lives of customers in various areas. Despite the development and significance of online grocery shopping, little is known about how individuals buy goods online. Khan et al. (2020)

Grocery shopping in the 21st century is changing drastically, and one major element of this change is online grocery shopping (Peregrin, 2018). Given this prior work, when juxtaposed with in-store shopping, online grocery shopping has the potential to dramatically limit the impact of both the cognitive barriers to healthy food access as well as community access barriers related to healthy food purchase within the supermarket food environment: consumers can shop online at any time and online grocery shopping allows low-income food desert dwellers and customers with limited mobility to order groceries online and have them delivered (Appelhans, 2017).

Internet-based company is quickly becoming a recognized and used business strategy. More entrepreneurs are incorporating web-based commercial utilities and online exchange structures into their websites. So, it's not difficult to predict that within a couple of decades, the web will be the standard for purchasing any kind of product or administration. (Shuvo, 2018)

No firm can afford to ignore the functionality and new methods of conducting business that e-commerce provides. This application made advantage of business-to-consumer marketing (B2C). Online e-Commerce websites such as eBay provide a valuable platform for users to purchase and sell things. As a result, I decided to create a beneficial online grocery e-commerce platform that can be used by anyone and location bound free.

**1.3 Statement of the Problem**

The factors that influence the course of development of this work is the problem that customer encounter when they want to purchase groceries, customer have to practice in-store shopping, and some even go around with a lot of cash to purchase groceries which is very dangerous, all this problem motivate me to embark on this project work to eradicate the above problems mentioned. Thus, it is necessary to introduce an information system that would be used for the recording of events regarding and complications related to grocery ordering.

**1.4 Aim and Objectives of the Study**

The project is aimed at developing an e-commerce grocery platform that can be used by anyone interested in purchasing groceries. Which will efficiently meet customers’ grocery demands and ensure proper financial accountability.

**Objectives:**

In other to achieve the aim of this project the following objectives are set and considered relevant for the achievement. This includes:

1. To reveal the related literature on E-commerce grocery business to customer.
2. To design E-commerce grocery business to customer and to meet customer transactions needs.
3. To implement the system and evaluate his efficiency in terms of system information needs / output.

**1.5 Scope of the Study**

This research work will concentrate on an e-commerce online grocery store, viewing products, adding to a cart, purchasing, payment of the product, and issuing of receipt. This research work will not go beyond.

**1.6 Limitations of the Study**

Some of the limitations that may have influenced the conclusion of this research include:

Due to the school academic calendar, the time window allocated to complete this project was quite short, and it was carried out under pressure, causing the researcher to fail to include several required elements.

The requirement for a typical functional personal computer unit to execute and debug the application software hampered the task's quick and simple progress

The lack of a steady, efficient, and dependable power source was another key constraint for this project, as it suspended the work multiple times during the design and testing stages.

**1.7 Significance of the Study**

The main importance of this study is to alleviate the problems and stress that customers have during grocery shopping by keeping track of stock lists, customer lists, sales volume, and summary reports.

**1.8 Project Organization**

For ease of study and proper understanding of this project write-up, it is planned and organized into five chapters as follows:

Chapter one contains an introduction to the write-up, the problem statement of the study, the aims and objectives of the study, the significance of the study, the scope and limitation of the study, and the organization of the report. Chapter two contains the literature review of the study, an overview of the proposed system, and the contribution of other scholars on the subject matter being discussed. Chapter three presents the research methodology used in the development of the system to facilitate an understanding and effective future implementation of the system and also the presentation of the results of system analysis and design. Chapter four contains system design implementation and documentation, design of the system, output design, input design, and system requirements for implementation and lastly Chapter five provides a summary of major findings, conclusions, and recommendations based on the study conducted.

**1.9 Definition of Terms**

1. **E-commerce**: E-commerce, or electronic commerce, refers to the buying and selling of goods or services over the Internet.
2. **Shopping Cart:** asoftware application that allows an online shopper to accumulate a list of items for purchase
3. **Merchant:** A merchant is a person or company that sells goods or services.
4. **B2C:** business-to-consumer, refers to a type of e-commerce transaction in which a business sells goods or services directly to a consumer.
5. **Online grocery shopping:** refers to the process of purchasing groceries and other household items through the Internet. This can be done through a retailer's website or app, or a third-party online grocery delivery service.
6. **User Interface (UI):** The visual elements and layout through which users interact with a software application.
7. **vii. Backend:** The behind-the-scenes operations and data processing of a software application.
8. **viii. Responsive Design:** Creating web interfaces that adapt and function well across various devices and screen sizes.
9. **ix. Database Management System (DBMS):** Software used to manage and organize data efficiently.
10. **x. User Experience (UX):** The overall quality and satisfaction users derive from interacting with a product or service.

**CHAPTER TWO**

**LITERATURE REVIEW**

**2.1 Introduction**

This chapter seeks to show how the topic under investigation links to earlier research, current practice, or other fields of knowledge by citing relevant publications by other researchers that have addressed a related issue. Furthermore, this chapter will give a synthesis of current research on the topic, noting areas of agreement, disagreement, and gaps in the literature, to demonstrate the project topic's importance in the field and to identify prospects for further research.

**2.2 E-Commerce Grocery Business to Customer**

Smart Online Grocery Shopping App Development. Economic growth contributes to the worldwide spread of e-commerce by increasing demand for online services, which increases the number of providers and the size of their delivery networks. People have changed their focus from markets to apps to restrict the transmission of the COVID-19 virus during the pandemic. Most stores, malls, and markets use their applications. The research gaps were filled in this work by building and evaluating a conceptual model for online grocery shopping as well as the aspects that influence the customer's attitude. Then, create clever mobile apps that can reply to all consumer requirements and e-commerce features like an expert system. (Abdulhaseeb, 2021)

Furthermore, the system used clever data mining to collect client history and aggregate items depending on consumer wants. It gives updates about new goods and special discounts. The Flutter programming language and the Android studio tools are used to develop the suggested mobile application for online grocery shopping that is compatible with many operating systems (Android and IOS). (Abdulhaseeb, 2021)

In conclusion, the protective situation produced by Covid-19 enforced a mobility ban and the necessity to distance. The necessity of having a mobile app for sale over the Internet that provides a quick and safe method to purchase and reduces the hassle of transferring between markets becomes clear here. (Abdulhaseeb, 2021)

Design and Implementation of an Online Grocery Store. E-commerce has grown in popularity as the internet has grown in popularity. The fundamental goal of e-commerce is to deliver a convenient and better buying experience. It has not only displaced conventional shopping but also brought convenience and ease of purchasing from your workstation. Consumers no longer have to visit stores at specific times; they may buy practically anything from anywhere at any time. Continuing this trend, there is a need for an online grocery shop since it can improve the present system even more. Consumers no longer need to go to a local food shop; they can get anything while sitting at home at any time. It is quick, easy, and adaptable. It offers multi-vendor shopping in a single window. (Zikra, 2017)

Moreso, according to Zikra (2017) she stated that the web shop may be built with a variety of technologies, including PHP, JavaScript, and JSP. The basic architecture of websites is built using HTML/CSS and PHP. MySQL is used to store customer and vendor information (sellers).

In conclusion, the built online store will allow users to buy grocery items from a broad variety using a secure payment method. It will improve the customer experience and encourage a hassle-free purchasing environment. (Zikra, 2017)

Online System Based on E-Commerce Platform. E-commerce is now well-known around the world. The issue arises when users (customers) have to come to the store at a specified time since most offline stores have restrictions on the opening time, and additionally, people are busy nowadays with their own lives and do not have time to go to the store at a given time. Finally, due to geographical constraints, an offline business cannot be conducted from any place. As a result, the online shop, whether by web or mobile application, offers a more convenient shopping experience due to the diversity of items, infinite time to shop, and accessibility at any time and from any location. (Nurfarah, 2018)

Furthermore, the Waterfall Model was the first process model developed, as well as the first SDLC technique utilized for software development, which is why this model is employed to create this project due to its simplicity. Small programs known as units are created during the system design process. This project is built with SQL server, HTML, CSS, and PHP. During the design phase, the system is built around the database and interface design. (Nurfarah, 2018)

To summarize, E-Commerce has become an important resource in a modern company, not only from the standpoint of the entrepreneur but also from the standpoint of the client. This project provides the customer with an e-commerce website where they can sell or purchase valuable used items over the internet, making it easier and more convenient for the user. (Nurfarah, 2018)

Implementation of an E-Shopping Website. Many small and medium-sized business owners in Bangladesh are unaware of e-commerce web applications. This can help them build their business, but it requires upkeep, security updates, and a lot of added functionality. This project's purpose is to expand a modern dynamic e-commerce internet software as an online or digital store on the internet. Where one-of-a-kind kinds of products can be sold from the comfort of one's own home over the internet. (Shuvo, 2018)

Moreso, the researcher chose certain technologies and frameworks to improve the process. HTML, CSS, JavaScript, Jquery, and PHP are some of the technologies used in this project. The research employed MySQL as a relational database management system (RDBMS) for database design. For local host development, the research used XAMPP. The newest modern PHP framework Laravel was employed in the study for the framework (Version 5.7) (Shuvo, 2018)

In conclusion, the project provided the researcher with a better grasp of the development process and technology utilized to construct a modern dynamic eCommerce web application. The project's design comprises the data model and workflow with real-world examples. The project's development has provided the researcher with detailed knowledge of the most recent technology, such as "Laravel," for developing an eCommerce web application, as well as improved knowledge of the eCommerce business industry. (Shuvo, 2018)

Development of an E-Commerce-Based Online Web Application for the COVID-19 Pandemic. Daily, people must visit a grocery store to obtain necessities. To obtain the necessary products, individuals go to a grocery store, select items from various shelves, pay for the item, and leave. However, this technique may not always appear to be extremely convenient. People these days are extremely busy. They usually neglect to make time for grocery shopping because of their hectic schedule. Some individuals attempt to avoid it due to traffic, merely to escape crowds. Furthermore, after considering all of the challenges and causes of the troubles, we believe that an online grocery shop is an ideal answer to all of the problems. (Mohammad, 2020)

Furthermore, the researchers developed an "Online Grocery Shop" online application for everyone to make shopping easier, safer, more entertaining, and more effective while also saving time. There are two user profiles in the system: a customer account and an admin account. Customers may order things using a valid Gmail address and a secure password. They can confirm their order after completing the registration procedure successfully. (Mohammad, 2020)

Finally, the method can make people's lives easier, more convenient, and more pleasant, as well as have an impact on Bangladesh's economy. In exchange, internet shopping has allowed many small shops to enter the market that would not have been able to do so if they had to bear the hefty costs of establishing a brick-and-mortar store. In the end, it was a win-win situation for both the buyer and the seller. (Mohammad, 2020)

**2.3 Summary of Related Literature Reviews**

|  |  |  |
| --- | --- | --- |
| **Author & Year** | **Title & Description** | **Merit and Demerits** |
| Abdulhaseeb et al. (2021). | Smart Online Grocery Shopping App Development.  The system (online grocery Shopping) is a multistage smartphone application that assists users in shopping and delivering the necessary supplies fast and safely. | The system is portable, and not dependent on any operating system.  Products are not well classified into their respective categories |
| Zikra et al. (2017). | Design and Implementation of an Online Grocery Store.  The system's goal was to create an online grocery shop with a simple and easy-to-use interface and safe transactions. | The system provided a secured payment for its users.  The system may lack scalability. |
| Nurfarah (2018). | Online System Based on E-Commerce Platform.  The Online System Based on an E-Commerce Platform is a system designed for selling and purchasing used items, similar to the websites. | The system provides an opportunity for customers to chat with sellers directly.  The system lacks shipment management. |
| Shuvo (2018). | Implementation of an E-Shopping Website.  This article proposes an implementation of an electronic shopping website that will be free of hassle for its users. | The system provided a rich user interface and ease of use  The system is facing security threats from its online payment. |
| Mohammad et al. (2020). | Development of an E-Commerce-Based Online Web Application for the COVID-19 Pandemic.  This study focuses on providing a basic platform for  startup of an online grocery shop | The system made people's lives easier, more convenient, and more pleasant.  The system is limited to the web. |

**CHAPTER THREE**

**METHODOLOGY AND DESIGN**

**3.1 Introduction**

A methodology is a formal study or research, particularly to unveil new facts or information; thus, research methodology should be good enough to enable the accomplishment of the specified objectives, which are accomplished through the use of specific components such as data collection and design procedures, and system modeling (use case, activity, and class diagrams). This chapter describes the input/output specifications as well as the system requirements for the e-commerce grocery platform that is currently being built.

**3.2 Methods of Data Collection**

Before constructing any system, it is necessary to collect data and facts about the existing system to comprehend what is going on. Two approaches were used in this study.

**3.2.1 Observation of the Work Environment**

The strategy employed to gather valuable information and data for this study involved a meticulous examination of the operational dynamics within the confines of Sammy Groceries Supermarket FCT Abuja. The observational approach was adopted to discern the intricacies of the prevailing manual system in action. This method facilitated an in-depth exploration of the supermarket's daily processes and exposed the most conspicuous shortcomings within the existing system.

The observation was conducted within the distinctive setting of Sammy Groceries Supermarket, where the intricacies of the manual operations were closely observed and analyzed. The physical environment of the supermarket was meticulously scrutinized, encompassing aspects such as customer interactions, inventory management, cash handling, and overall workflow. The observatory exercise allowed for an unobtrusive yet comprehensive understanding of the supermarket's modus operandi.

The observation of the work environment at Sammy Groceries Supermarket served as a vital investigative tool that enabled a comprehensive analysis of the existing manual system. By immersing within the supermarket's daily operations, this approach unveiled the operational intricacies and highlighted the areas necessitating transformation.

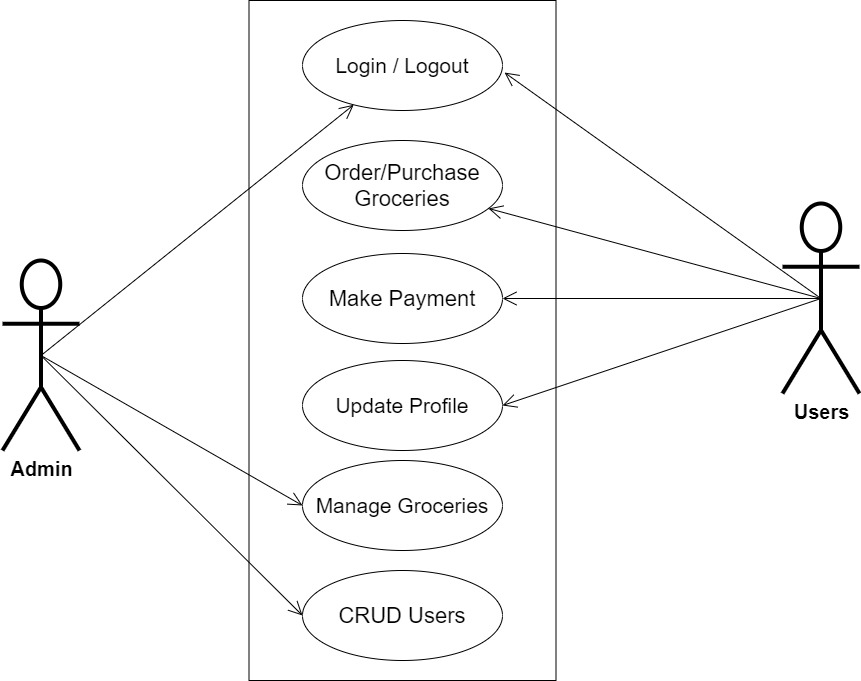
**3.2.2 Documentation**

Secondary data gathering includes documentation. Journals, books, previous work, publications, and other sources are used in this manner. This data-gathering technique is chosen because it allows for comparison with previous research. This includes the internet, which is a tool for data collection. The internet was utilized to study complex or unclear problems.

**3.3 System Modeling**

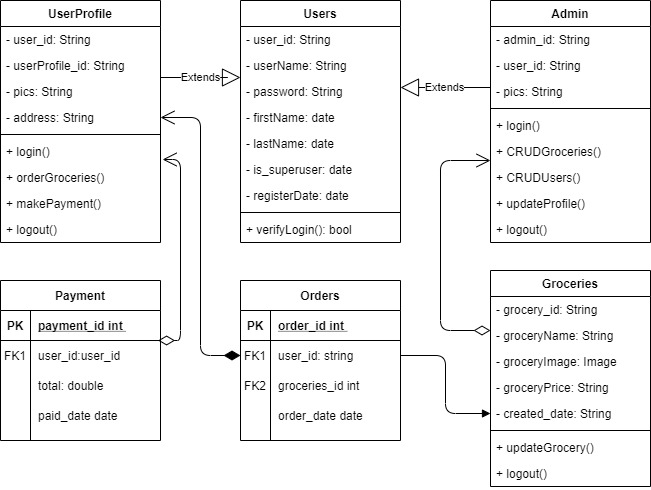
A system model is a conceptual model that characterizes and portrays a system as an outcome of system modeling. It is the connection of several components that collaborate to accomplish a shared goal. A collection of visual notation methods included in the Unified Modeling Language, which is utilized in the creation of this current system, may be used to generate visual models of object-oriented software-intensive systems. Use case diagrams, class diagrams, and activity diagrams are among the UML diagrams used in this new design.

**3.3.1 Use Case Diagrams**



**Fig 3.1 System Use Case Diagram**

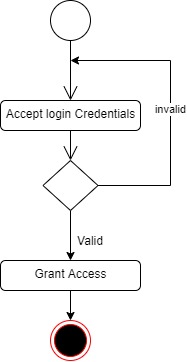
**3.3.2 Class Diagrams**



**Fig 3.2 System Class Diagram**

**3.3.3 Activity Diagrams**

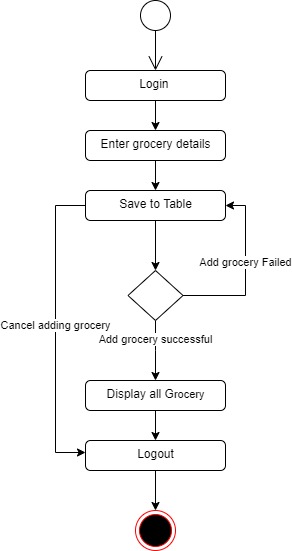
**Login**

The process for gaining access to the system is depicted in the diagram below; the username and password must be accurate to gain access.

**Fig 3.3 System Login Activity Diagram**

**Create Groceries**

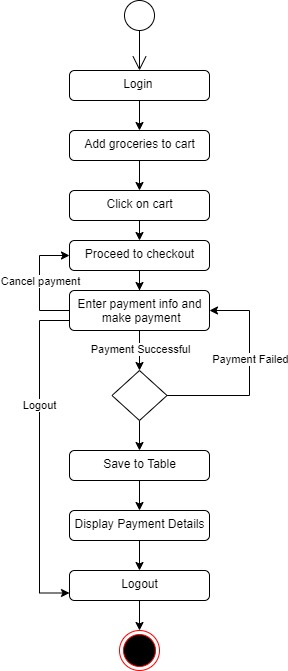
The process for adding groceries to the system is depicted in the diagram below; The system ensures that the user is authenticated and authorized to perform the creation.



**Fig 3.4 Create Groceries Activity Diagram**

**Make Purchase**

The process for adding to the cart and making payment for the groceries is depicted in the diagram below; The system ensures that the user is authenticated and authorized to perform the purchase for validity purposes.



**Fig 3.5 Make Purchase Activity Diagram**

**3.4 Database Design**

The following are some of the input specifications used in this project work.

1. Users Table: contains the generic information of all system users.
2. Groceries Table: contains every grocery registered on the system.

**Table 3.1 Users Input Specification Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Length** | **Description** |
| user\_id | Varchar | No | PK | 32 | Unique string for identifying users |
| username | Varchar | No |  | 20 | Unique name for users |
| password | Varchar | No |  | 128 | User Password |
| first\_name | Varchar | No |  | 20 | User first name |
| last\_name | Varchar | No |  | 20 | User last name |
| phone | Varchar | No |  | 11 | User phone number |
| email | Varchar | No |  | 100 | User email address |

**Table 3.2 Groceries Input Specification Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Length** | **Description** |
| grocery\_id | Varchar | No | PK | 32 | Unique string for identifying citizens |
| groceryName | BigInt | No |  | 10 | Grocery name |
| groceryImage | BigInt | No |  | 60 | Image to identify the grocery |
| groceryPrice | Varchar | No |  | 60 | Price of the grocery |
| created\_date | Varchar | No |  | 60 | Date the grocery was registered |

**3.5 Output Design**

This declares and displays the outcome of the given input. This system's output is dependent on its input. The output specification is listed below.

**Table 3.3 Users** **output design table**

**e-COMMERCE GROCERY BUSINESS TO CUSTOMER APPLICATION**

**List of the System Registered Users**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **User\_id** | **Username** | **First\_name** | **Last\_name** | **Phone** | **Email** |
| XXXX | XXXX | XXXX | XXXX | XXXX | XXXX |
| XXXX | XXXX | XXXX | XXXX | XXXX | XXXX |
| XXXX | XXXX | XXXX | XXXX | XXXX | XXXX |
| XXXX | XXXX | XXXX | XXXX | XXXX | XXXX |

**Table 3.4 Groceries** **output design table**

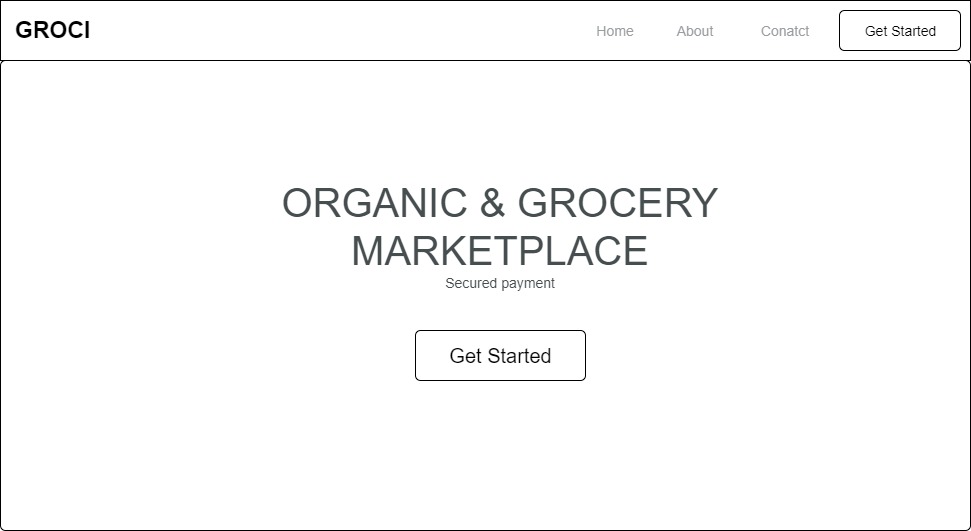
**e-COMMERCE GROCERY BUSINESS TO CUSTOMER APPLICATION**

**List of the System Registered Groceries**

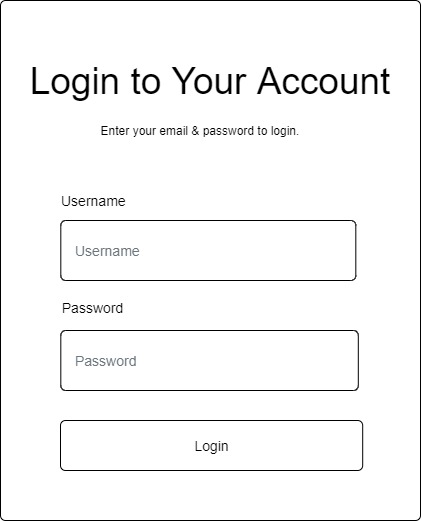
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Grocery\_id** | **GroceryName** | **GroceryImage** | **GroceryPrice** | **Created\_date** |
| XXXX | XXXX | XXXX | XXXX | XXXX |
| XXXX | XXXX | XXXX | XXXX | XXXX |
| XXXX | XXXX | XXXX | XXXX | XXXX |
| XXXX | XXXX | XXXX | XXXX | XXXX |

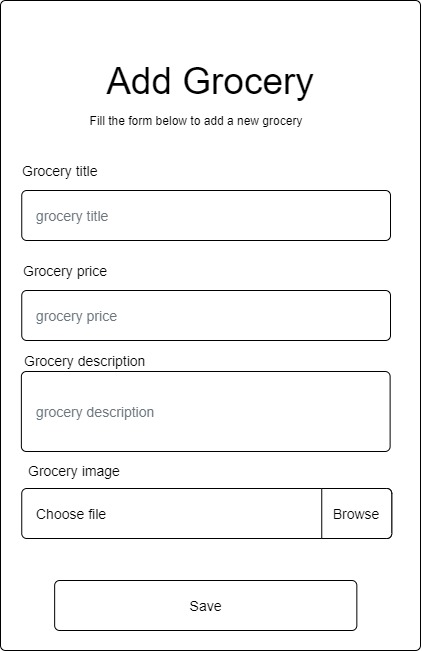
**3.6 Input & User Interface Design**

This is a graphic depiction of the system interface; it will be designed to be user-friendly, responsive, and visually beautiful. Furthermore, it will be appropriately safeguarded, thus authorization will be required to see certain levels of the information. To help with the designs, a mid-fidelity wireframing program called Draw.io is employed.



**Fig 3.6 Home Page**



**Fig 3.7 Login Form**

**Fig 3.8 Add Grocery**

**3.7 System Requirement**

Every software system built has a stated system requirement on which it is meant to execute for the best performance. The system requirements, on the other hand, are the bare minimum of hardware and software required for the system to work properly.

**3.7.1 Hardware Requirement**

System Hardware Requirement Include:

1. Minimum of 2 GB of RAM (Random Access Memory).
2. Minimum of Intel Dual core processor.
3. Minimum of 250GB HDD (Hard Disk Drive).

**3.7.2 Software Requirement**

The software requirements include:

1. At least windows 7 OS (Operating System).
2. Vs. Code IDE installation.
3. Browsers include Chrome and Firefox.

**3.8 Choice of Programming Language**

This research project will be a web-based application built on a relational database architecture (SQLite). For frontend development, HTML (hypertext markup language), CSS (cascading style sheet), and JavaScript will be used, while Django (Python) will be used for backend programming. The combination of the above will help build a very robust eCommerce grocery business-to-customer application that will be useful, fast, and handy.

**CHAPTER FOUR**

**SYSTEM IMPLEMENTATION EVALUATION**

**4.1 Introduction**

The implementation process of the new system is thoroughly described in this section, emphasizing its efficiency and effectiveness. It provides real-life examples of how the system functions and outlines the sequential steps required for its successful implementation.

* 1. **System Testing and Evaluation**

Conducting thorough testing of the developed system is essential for multiple reasons. One significant objective is to identify and address any potential shortcomings or defects in the system. In this project, a blend of unit and integration testing was utilized to validate the design's efficacy and efficiency, guaranteeing that the new system meets its functional requirements and operates without any errors.

**Unit Testing**

This part examines specific units or single components of the system individually to confirm that specific phases function properly and without problems.

**Integration Testing**

Integration testing was performed on the software, wherein all components were brought together and operated as a unified system. The objective of this testing was to validate the connectivity and proper integration of the various parts, ensuring seamless collaboration among the units.

**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**

Given that the website has a public scope, certain information such as the index page and login page are accessible to all visitors. However, there are specific functionalities and information that are restricted and not available to every visitor. To enforce these restrictions, passwords are implemented, granting different levels of access to users. The administrator holds the highest level of access, followed by users with lesser privileges. This approach ensures that certain features and data are only accessible to authorized individuals while maintaining the overall public availability of the website.

**4.5 Sample Outputs**

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

**Homepage**

The page serves as a gateway to navigate and explore the various sections of the website.

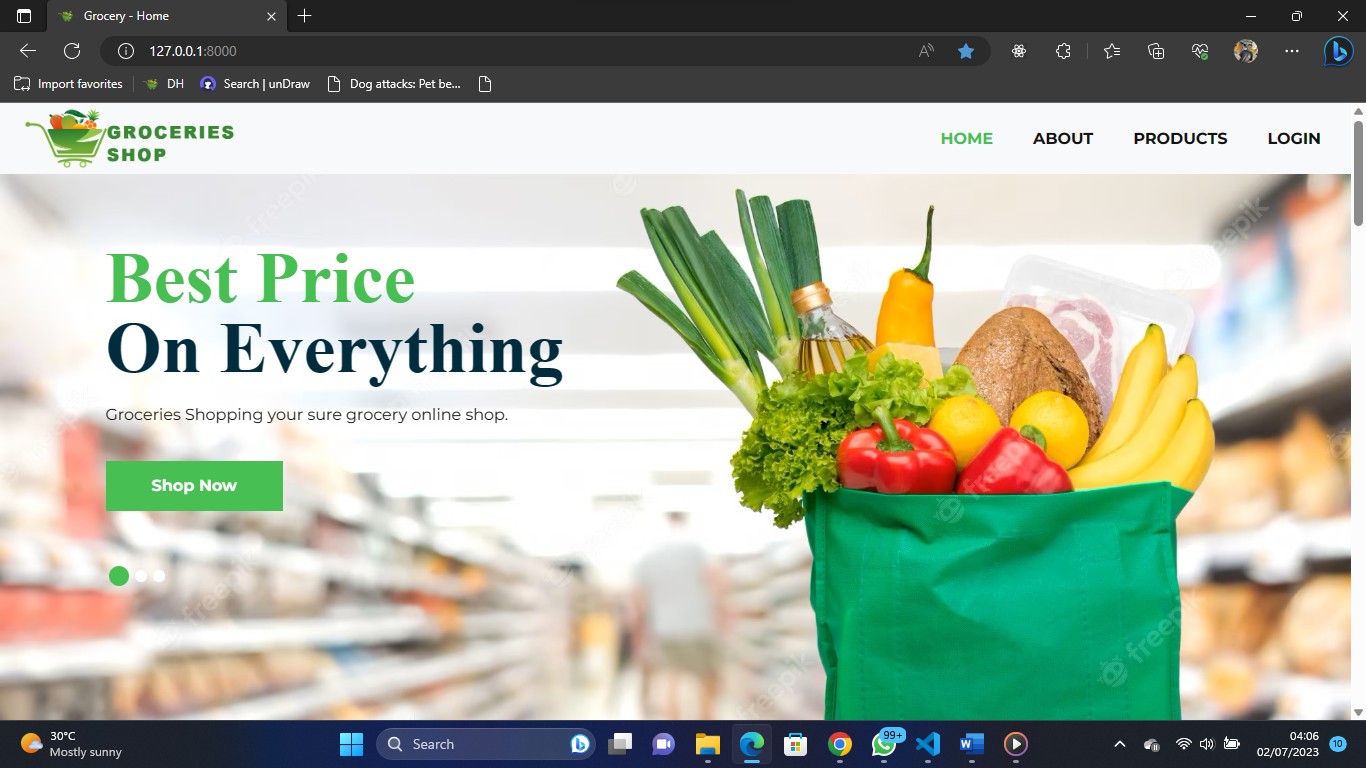


Fig 4.1 Homepage

**Customer Login**

This is a page that grants users access to the system only if the correct credentials are provided.

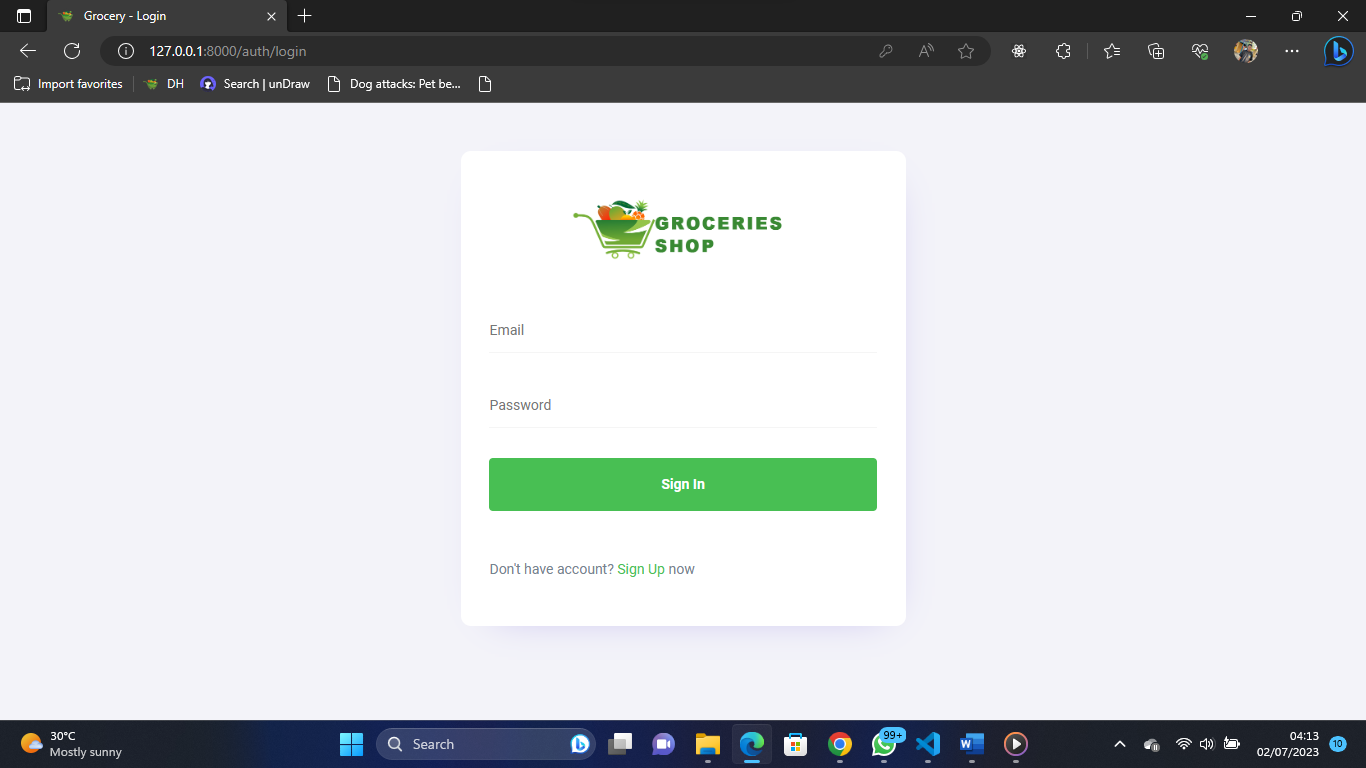


Fig 4.2 Customer Login

**Customer Registration**

This is where the customer can register by providing the right credentials.

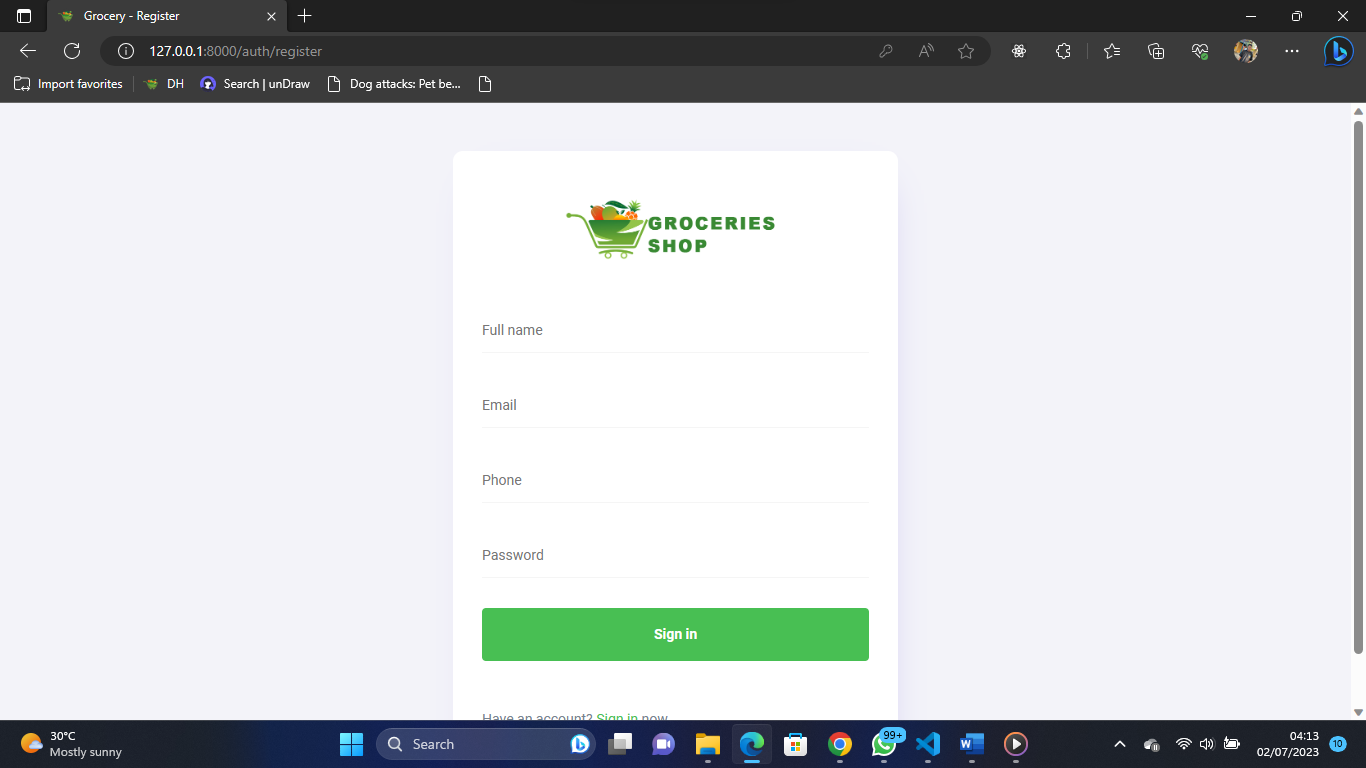


Fig 4.3 Customer Registration

**Customer Dashboard**

This is the customer dashboard, the sidebar shows the available functionality for the customer

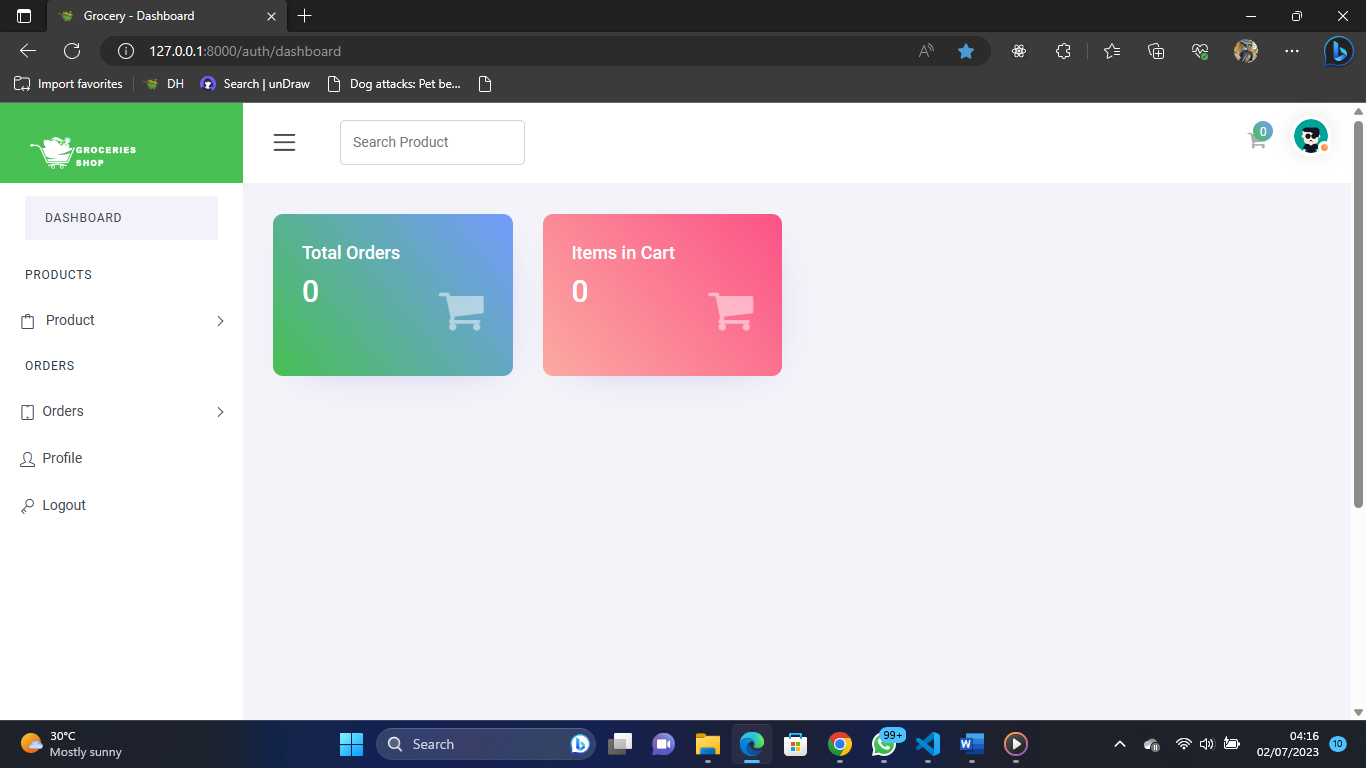


Fig 4.4 Customer Dashboard

**Available Groceries**

This is the page where the customer can see all available groceries and make purchases.

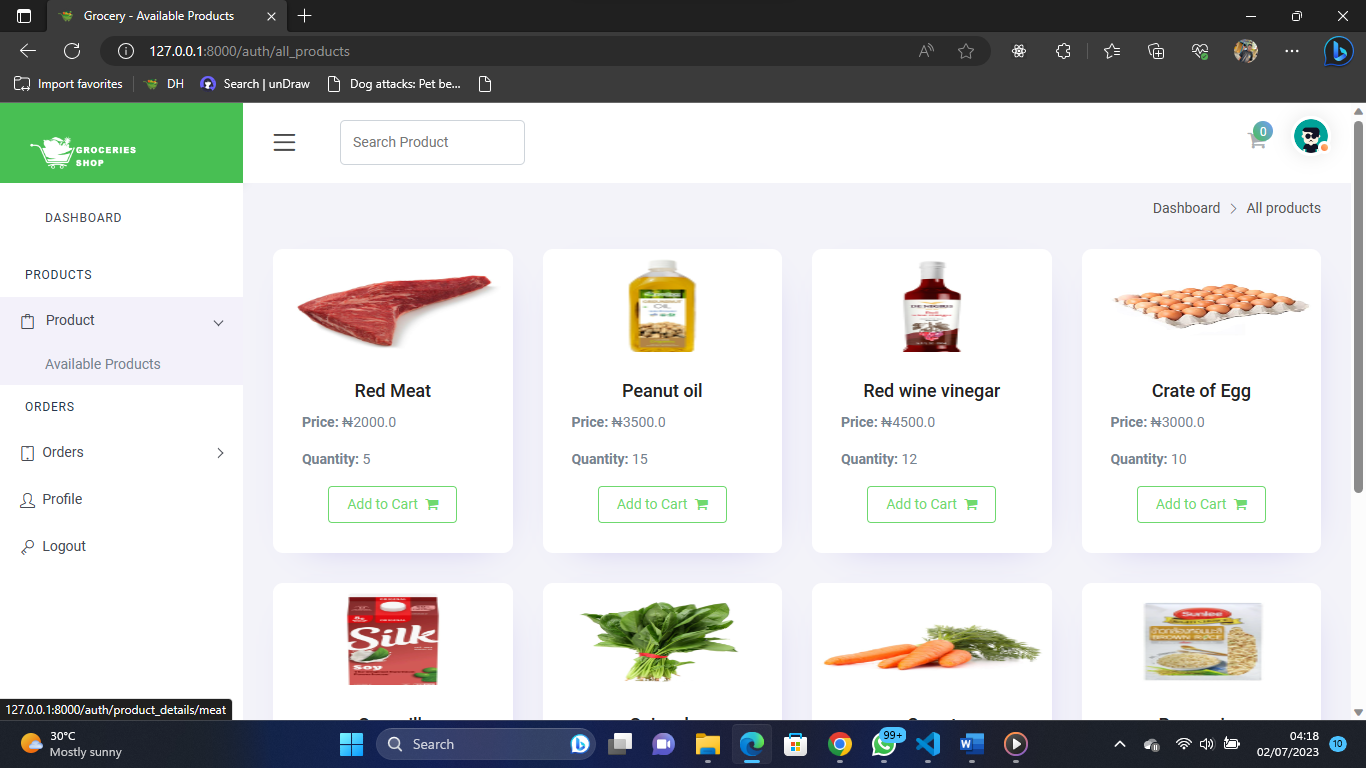


Fig 4.5Available Groceries

**Items in Cart**

This is where the customer can view the items in the cart and decide to checkout

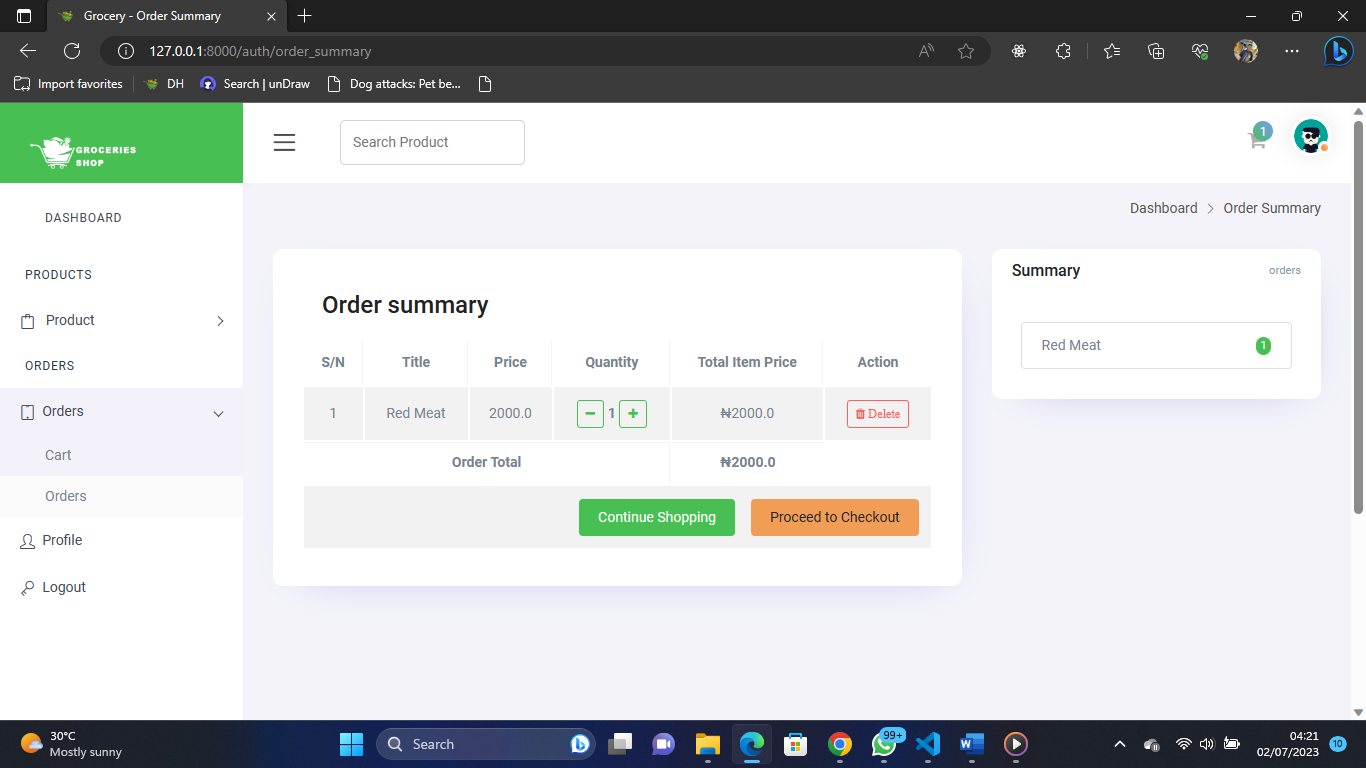


Fig 4.6 Items in Cart

**Checkout Form**

This is where the customer can make payment for the items in the cart

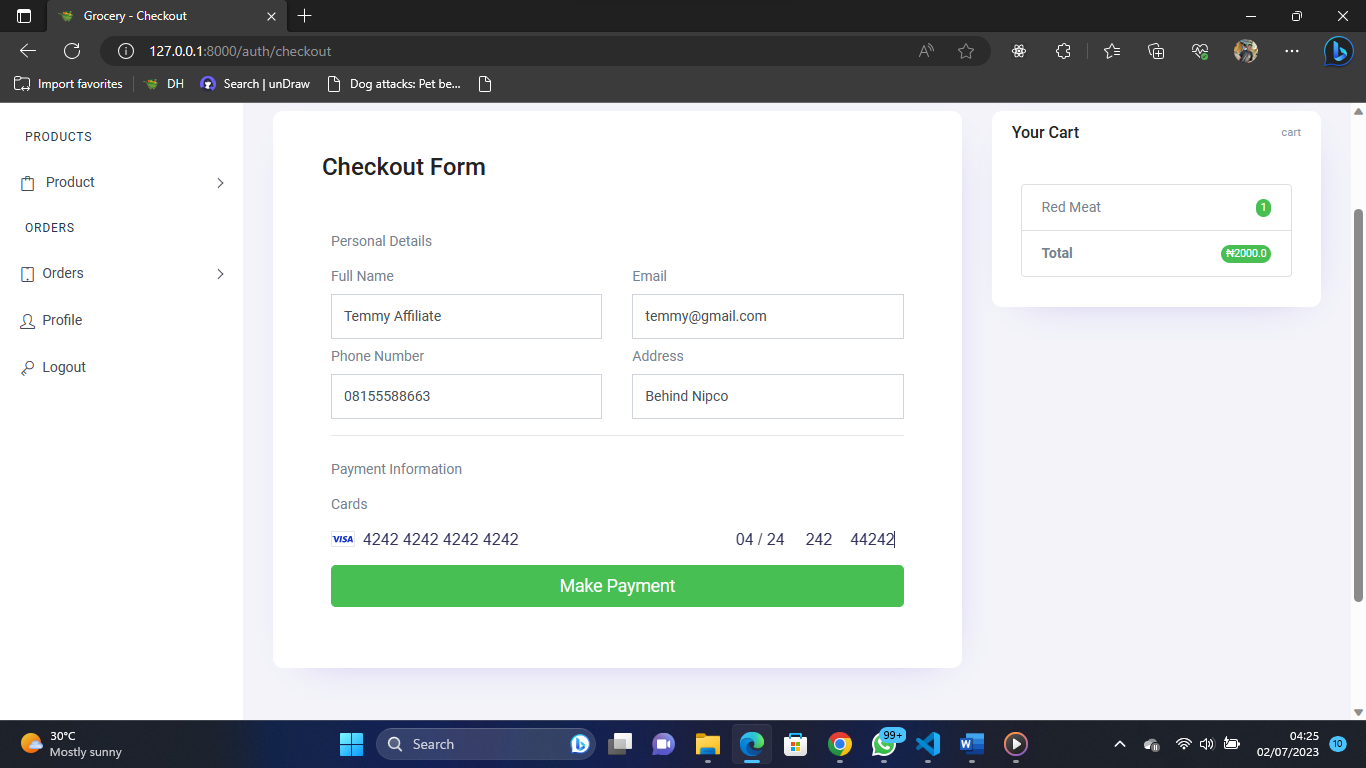


Fig 4.7 Checkout Form

**Customer Profile**

This is the customer profile page where the customer can view, and change account information

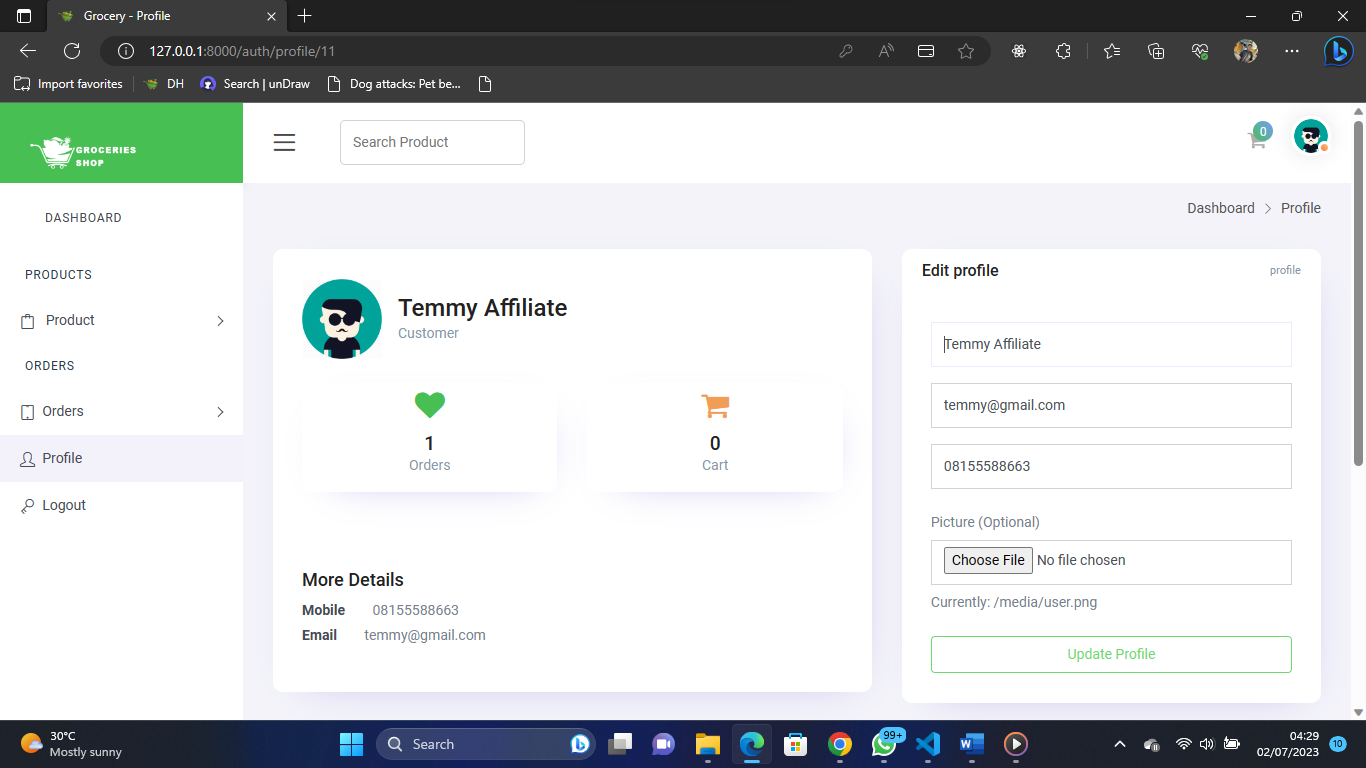


Fig 4.8 Customer Profile

**Payment Receipt**

This is the payment receipt received when customer have successfully made payment

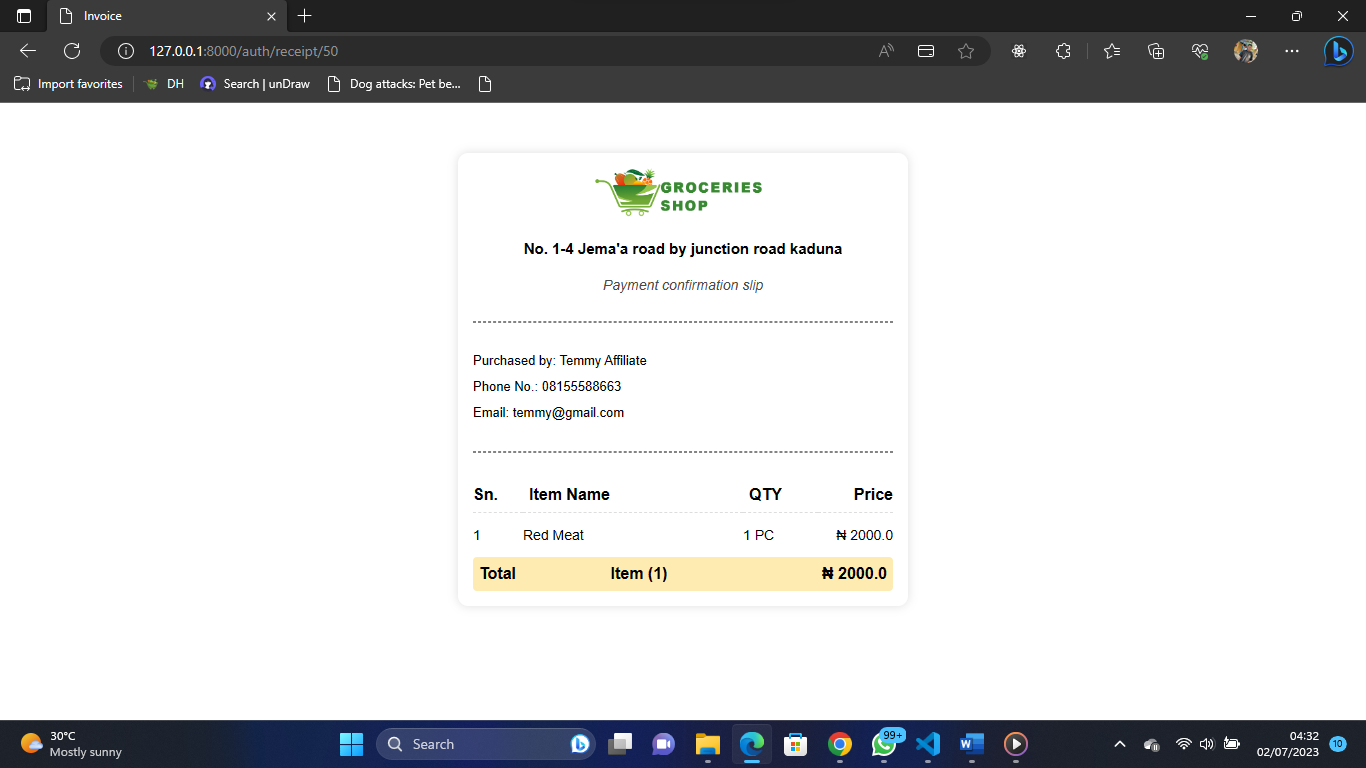


Fig 4.9 Payment Receipt

**Admin Dashboard**

This is the admin homepage, the sidebar shows the available functionality for the admin

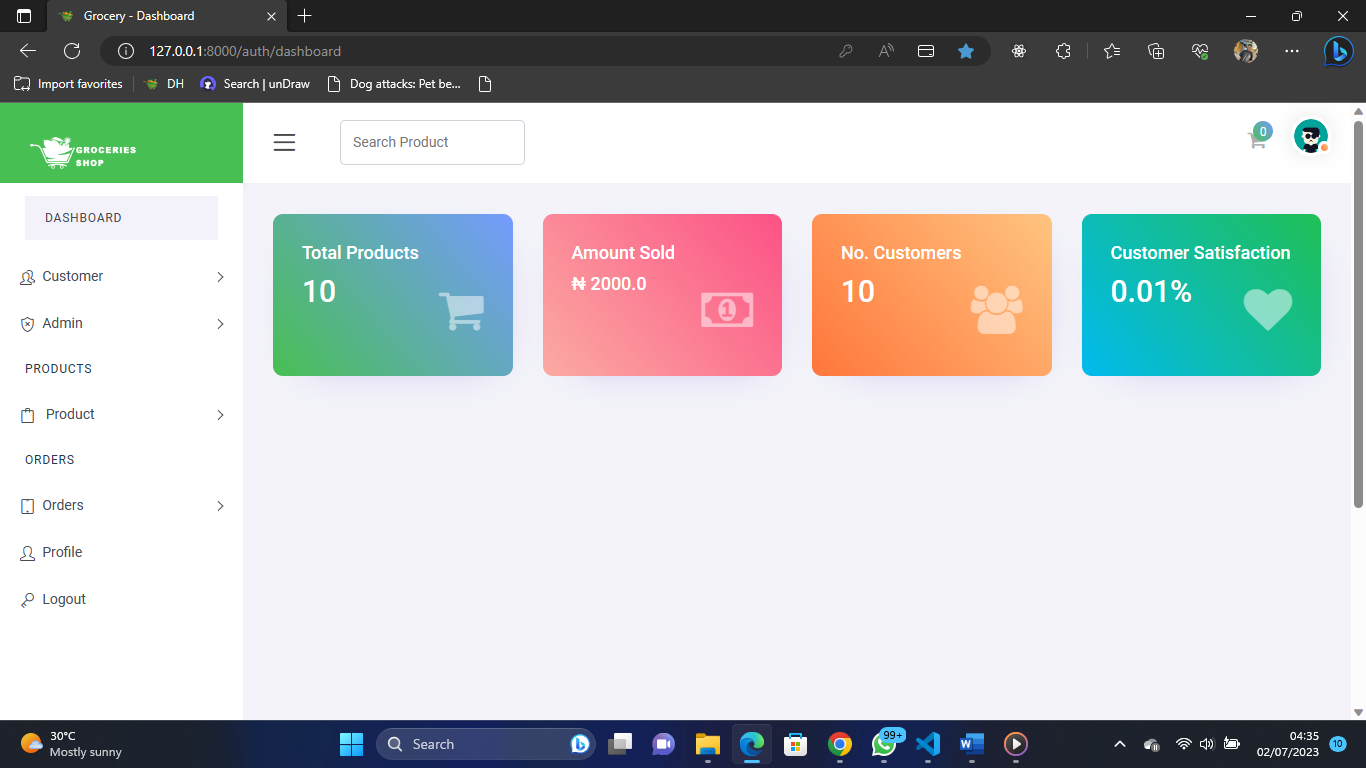


Fig 4.10 Admin Dashboard

**Add Product**

This is where the admin can add newly available groceries to stock

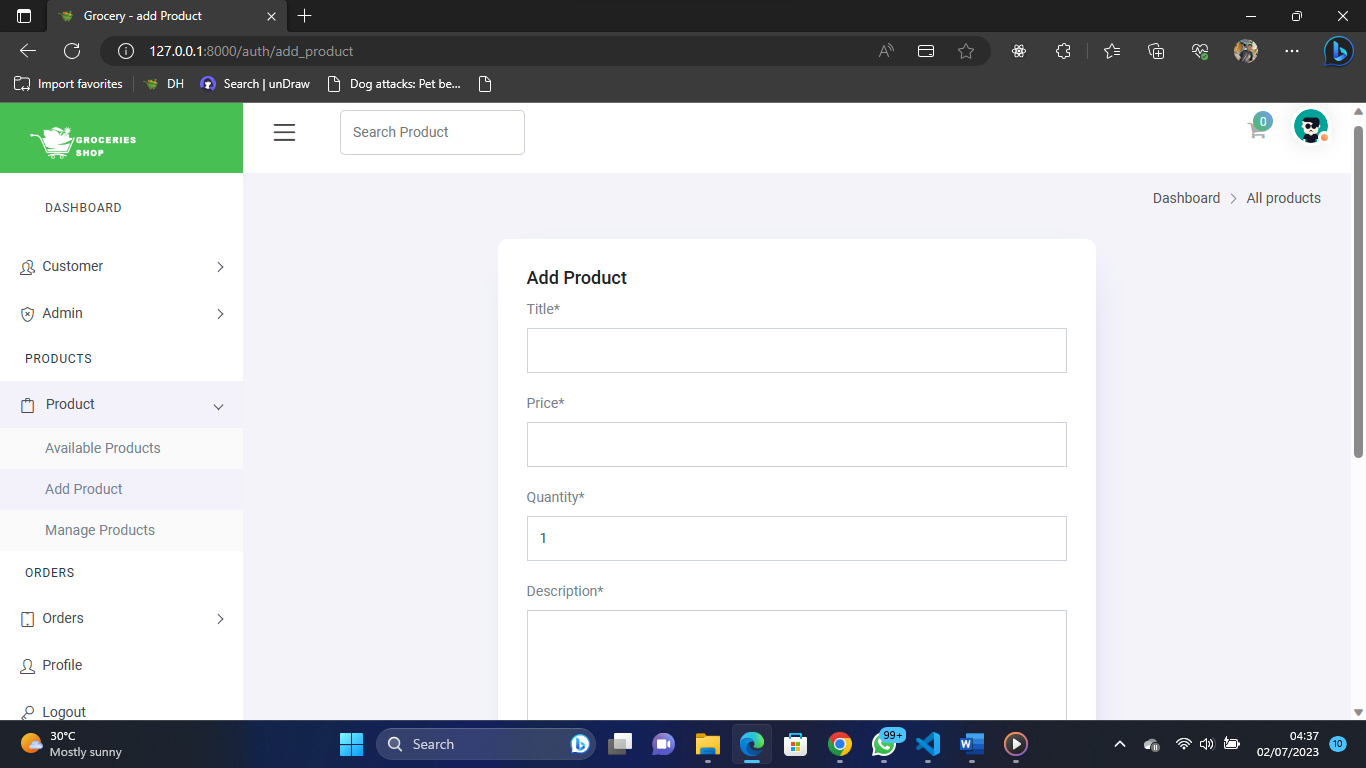


Fig 4.11 Add Product

**Manage Groceries**

This is the page where the admin modifies or makes changes to the available groceries.

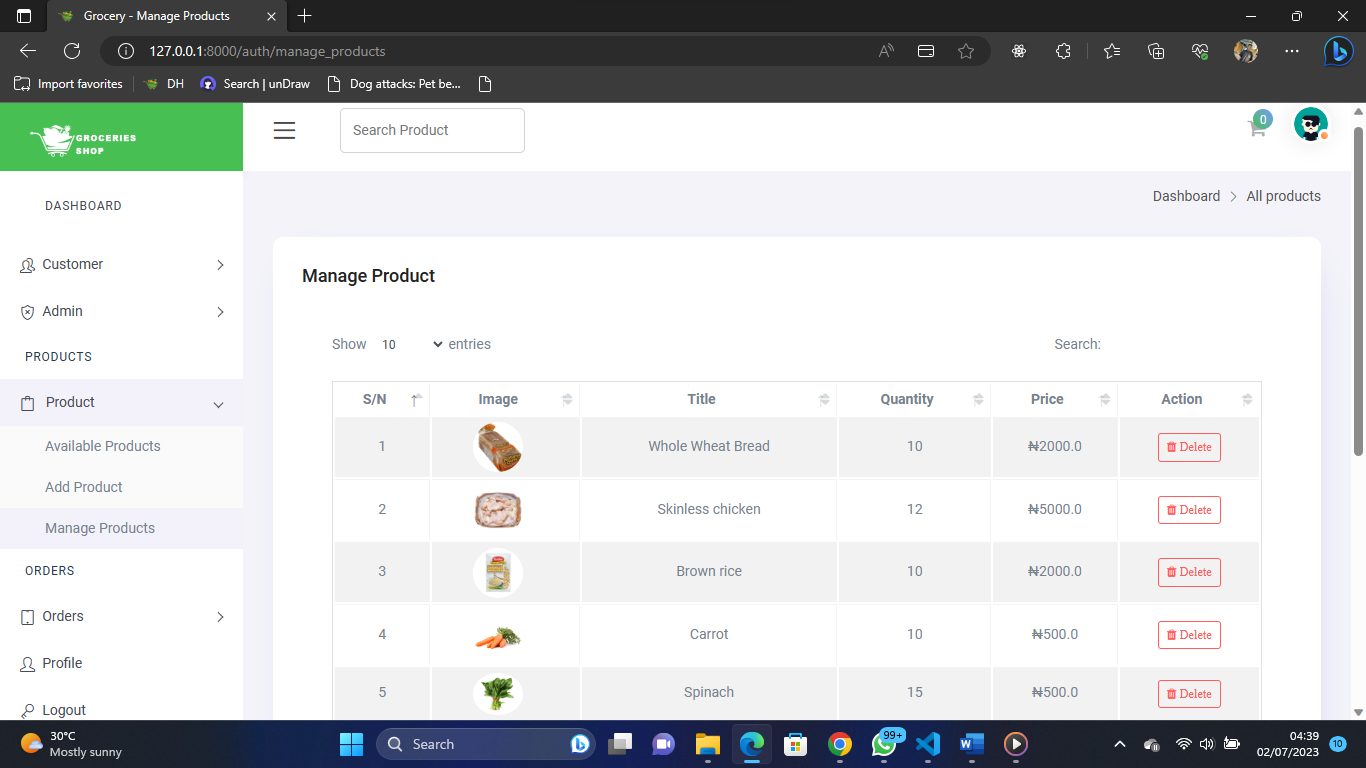


Fig 4.12 Manage Groceries

**Manage Orders**

Admin can use this page to view and confirm delivery of customer’s orders

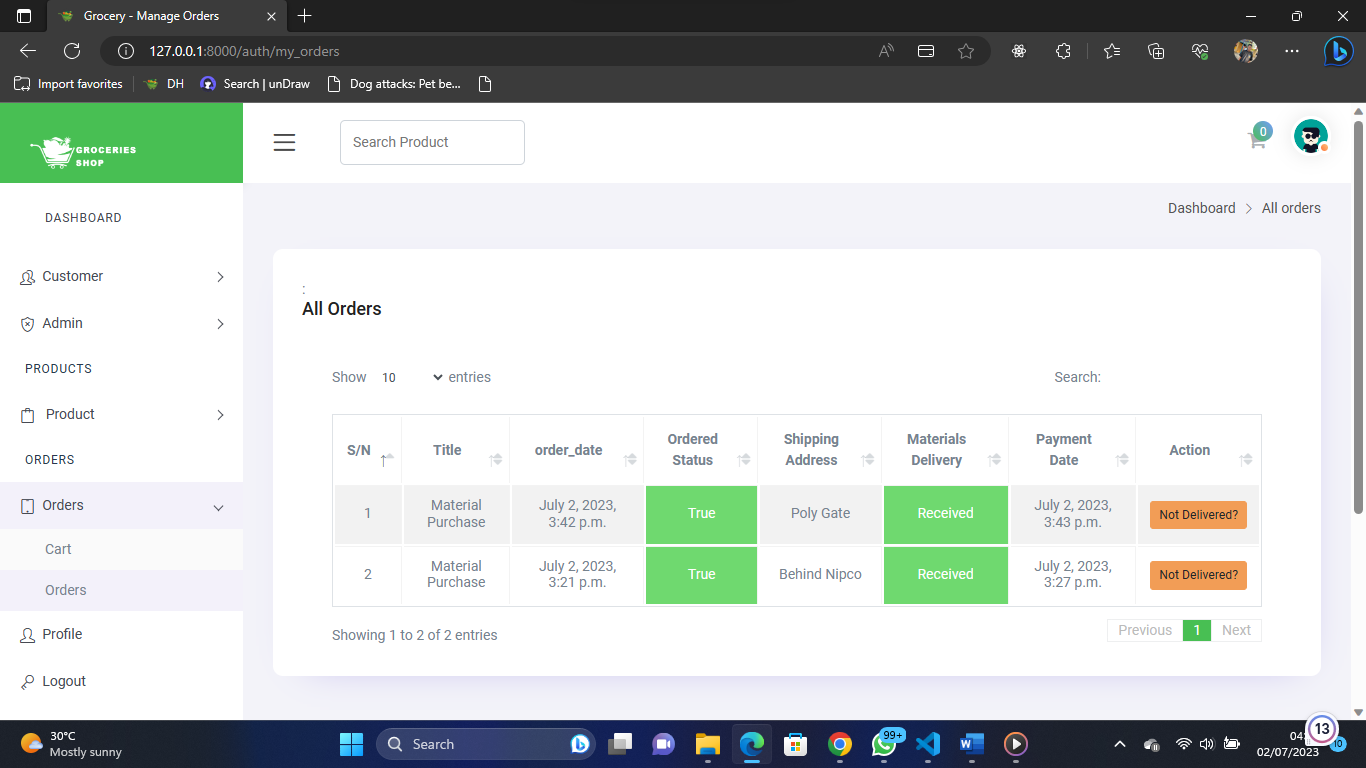
****

Fig 4.13 Manage Orders

**CHAPTER FIVE**

**SUMMARY CONCLUSION AND RECOMMENDATION**

**5.1 Summary**

The rise of online and mobile technology has transformed grocery shopping, prompting a shift towards online platforms. This project aims to develop an e-commerce grocery platform that offers convenience and efficiency to customers. The system will allow users to browse products, add items to their cart, make secure payments, and receive detailed receipts. By addressing the limitations of traditional grocery shopping, such as time constraints and safety concerns, this platform aims to enhance the overall shopping experience. The project holds significance in providing effective inventory management, customer tracking, and comprehensive sales reports.

**5.2 Conclusion**

In conclusion, the development of an e-commerce grocery platform for online shopping presents a promising solution to address the challenges faced by customers in traditional grocery shopping methods. This platform aims to provide convenience, accessibility, and improved safety for customers. The inclusion of features such as inventory management and detailed receipts enhances transparency and accountability in the purchasing process. Overall, this project lays the foundation for a modern and efficient approach to grocery shopping, offering potential benefits to both customers and retailers in the realm of online commerce.

**5.2 Recommendation**

Based on the findings and implementation of the e-commerce grocery platform, the following recommendations are proposed:

1. Expand Product Range: Regularly update and expand the range of products available on the platform to cater to a wider customer base.
2. Strengthen Security Measures: Implement robust security measures to safeguard customer information and financial transactions.
3. Customer Feedback and Support: Establish channels for customers to provide feedback, and suggestions, and report any issues they encounter while using the platform.

By implementing these recommendations, the e-commerce grocery platform can strive towards providing an exceptional shopping experience, attracting a larger customer base, and establishing itself as a trusted and preferred destination for online grocery shopping.

**REFERENCES**

Appelhans, BM, Lynch, EB, Martin, MA et al. (2017) Feasibility and acceptability of Internet

grocery service in an urban food desert, Chicago, 2011–2012. Prev Chronic Dis 10, E67.CrossRefGoogle Scholar

Khan, M. M., Shams-E-Mofiz, M., & Sharmin, Z. A. (2020). Development of E-Commerce-Based

Online Web Application for COVID-19 Pandemic. IBusiness, 12(04), 113–126. https://doi.org/10.4236/ib.2020.124008

Mohammed, M. K., Mahizebin, S., Zerin, A. S., (2020). “Development of E-Commerce-Based

Online Web Application for COVID-19 Pandemic” [*Department of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh*](https://www.scirp.org/journal/articles.aspx?searchcode=Department+of+Electrical+and+Computer+Engineering%2c+North+South+University%2c+Dhaka%2c+Bangladesh&searchfield=affs&page=1&skid=0)*.*DOI: [10.4236/ib.2020.124008](https://doi.org/10.4236/ib.2020.124008). https://www.scirp.org/journal/paperinformation.aspx?paperid=104377

Mohammed, J. Y., (2021). “Smart Online Grocery Shopping App Development”.

<https://www.researchgate.net/publication/352761822_Smart_Online_Grocery_Shopping_App_Development>

Peregrin, T (2018) Understanding millennial grocery shoppers’ behavior and the role of the

registered dietitian nutritionist. J Acad Nutr Diet 115, 1380–1383.CrossRefGoogle ScholarPubMed

Zikra. A., Shital, M., Navindas. G., and Nidhi S., (2016). “Grocery store”. Bharati Vidyapeeth

College of Engineering Navi Mumbai, India. <https://www.ijtra.com/view/design-and-implementation-of-online-grocery-store.pdf>

**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 Price

                              </span>

                              <br>

                              On Everything

                           </h1>

                           <p>

                              Groceries Shopping your sure grocery online shop.

                           </p>

                           <div class="btn-box">

                              <a href="{% url 'basics:product' %}" class="btn1">

                              Shop Now

                              </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>

                     Groceries Shopping, your sure online grocery shop.

                           </p>

                           <div class="btn-box">

                              <a href="{% url 'basics:product' %}" class="btn1">

                              Shop Now

                              </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 Service

                              </span>

                              <br>

                              On Everything

                           </h1>

                           <p>

                              Groceries Shopping your sure online grocery shop.

                           </p>

                           <div class="btn-box">

                              <a href="{% url 'basics:product' %}" class="btn1">

                              Shop Now

                              </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>

               <li data-target="#customCarousel1" data-slide-to="2"></li>

            </ol>

         </div>

      </div>

   </section>

   <!-- end slider section -->

</div>

<!-- why section -->

<section class="why\_section layout\_padding">

   <div class="container">

      <div class="heading\_container heading\_center">

         <h2>

            Why Shop With Us

         </h2>

      </div>

<!-- product section -->

<section class="product\_section layout\_padding">

   <div class="container">

      <div class="heading\_container heading\_center">

         <h2>

            Our <span>products</span>

         </h2>

      </div>

      <div class="row">

         {% include "partials/basic\_products.html" %}

      </div>

      <div class="btn-box">

         <a href="{% url 'basics:product' %}">

         View All products

         </a>

      </div>

   </div>

</section>

<!-- end product section -->

<!-- footer start -->

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

<!-- footer end -->

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

{% endblock %}

**Views.py**

from xhtml2pdf import pisa

# Payment

import stripe

stripe.api\_key = "sk\_test\_51L5Xs6GCAqCizi1RncjTC84yc0J7jaecLFB5gj07ZDNWCREFyEylsunXTltlQleL3lWzEcLsqIFCInvn6wGYu2Xa00cIHRZjMz"

# Create your views here.

class DashboardView(LoginRequiredMixin, View):

    login\_url = '/auth/login'

    def get(self, request):

        happy = Order.objects.filter(delivered=True).count() / 100

        context = {

            'customers':Accounts.objects.filter(is\_staff=False).count(),

            'happy':happy,

            'products':Product.objects.all().count(),

            'amount\_sold':sum([amount.amount for amount in Payment.objects.all()]),

        }

        return render(request,'auth/dashboard.html', context)

class RegisterView(SuccessMessageMixin, CreateView):

    model = Accounts

    form\_class = AccountCreationForm

    template\_name = 'auth/register.html'

    success\_message = "Account created successfully, you can now login!!"

    def get\_success\_url(self):

        return reverse("auth:login")

    def form\_valid(self, form):

        form.instance.set\_password(form.cleaned\_data.get('password'))

        form.instance.email = form.cleaned\_data.get('email').strip().lower()

        return super().form\_valid(form)

class LoginView(View):

    def get(self, request):

        context = {

            'next': request.GET.get('next', None)

        }

        return render(request, 'auth/login.html', context)

    def post(self, request):

        email = request.POST.get('email').strip().lower()

        password = request.POST.get('password')

        if email and password:

            # Authenticate user

            user = authenticate(request, email=email, password=password)

            if user:

                if user.is\_active:

                    login(request, user)

                    messages.success(request, f'You are now signed in {user.get\_name()}')

                    return redirect('auth:dashboard')

                else:

                    messages.warning(request, 'Account not active contact the administrator')

                    return redirect('auth:login')

            else:

                messages.warning(request, 'Invalid login credentials')

                return redirect('auth:login')

        else:

            messages.error(request, 'All fields are required!!')

            return redirect('auth:login')

class LogoutView(View):

    def post(self, request):

        logout(request)

        messages.success(request, 'You are now signed out!')

        return redirect('auth:login')

class ProfileView(LoginRequiredMixin, View):

    login\_url = '/auth/login'

    def get(self, request, user\_id):

        user = get\_object\_or\_404(Accounts, id=user\_id)

        orders = Order.objects.filter(user=request.user).count()

        form = EditAccountCreationForm(instance=user)

        context = {

            'form':form,

            'orders':orders,

            'user':user,

        }

        return render(request,'auth/profile.html', context)

    def post(self, request, user\_id):

        user = get\_object\_or\_404(Accounts, id=user\_id)

        form = EditAccountCreationForm(request.POST, request.FILES, instance=user)

        if 'profile' in request.POST:

            if form.is\_valid():

                form.save()

                messages.success(request, 'Profile updated successfully!')

                return redirect('auth:profile', user\_id)

            else:

                messages.error(request, 'Error updating Profile!')

            context = {

                'form':form,

                'user':user,

            }

        else:

            password1 = request.POST.get('password1')

            password2 = request.POST.get('password2')

            context = {

                'form':form,

                'user':user,

            }

            if password1 and password2:

                if password1 != password2:

                    messages.error(request, 'Passwords does not match!')

                    return redirect('auth:profile', user\_id)

                if len(password1) < 6 :

                    messages.error(request, 'Password too short, ensure at least 6 characters!')

                    return redirect('auth:profile', user\_id)

                user.set\_password(password1)

                user.save()

                messages.success(request, 'Password reset successful!!')

                if request.user == user:

                    return redirect('auth:login')

                if request.user.is\_superuser:

                    return redirect('auth:profile', user\_id)

                return redirect('auth:login')

        return render(request,'auth/profile.html', context)

class AllProductsListView(LoginRequiredMixin, ListView):

    login\_url = '/auth/login'

    model = Product

    paginate\_by = 8

    template\_name = "auth/all\_products.html"

    ordering = ['-id']

    def get\_queryset(self):

        return Product.objects.filter(quantity\_\_gte=1).order\_by('-id')

class ProductDetailListView(LoginRequiredMixin, DetailView):

    login\_url = '/auth/login'

    model = Product

    template\_name = "auth/product\_details.html"

class ManageProductsView(LoginRequiredMixin, ListView):

    login\_url = '/auth/login'

    model = Product

    template\_name = "auth/manage\_products.html"

    # def get\_queryset(self):

    #     return Product.objects.filter(quantity\_\_gte=1).order\_by('-id')

class EditProductsView(SuccessMessageMixin, UpdateView):

    model = Product

    form\_class = AddProductForm

    success\_message = "Product has been edited successfully!"

    template\_name = "auth/edit\_product.html"

    def get\_success\_url(self):

        return reverse("auth:edit\_product", kwargs={

            'slug':self.kwargs['slug']

        })

class AddProductView(LoginRequiredMixin, CreateView):

    login\_url = '/auth/login'

    form\_class = AddProductForm

    model = Product

    template\_name = 'auth/add\_product.html'

    success\_url = reverse\_lazy("auth:manage\_products")

class ProductDeleteView(LoginRequiredMixin, SuccessMessageMixin, DeleteView):

    login\_url = '/auth/login'

    model = Product

    success\_message = "Product deleted successfully!"

    def get\_success\_url(self):

        return reverse("auth:manage\_products")

class CreateAccountView(SuccessMessageMixin, CreateView):

    model = Accounts

    form\_class = AccountCreationForm

    template\_name = 'auth/add\_customer.html'

    success\_message = "Account created successfully!"

    def get\_success\_url(self):

        return reverse("auth:add\_customer")

    def form\_valid(self, form):

        form.instance.set\_password(form.cleaned\_data.get('password'))

        form.instance.email = form.cleaned\_data.get('email').strip().lower()

        return super().form\_valid(form)

class ManageCustomersView(ListView):

    model = Accounts

    template\_name = "auth/manage\_customer.html"

    def get\_queryset(self):

        return Accounts.objects.filter(is\_staff=False).order\_by('-id')

class AccountDeleteView(LoginRequiredMixin, SuccessMessageMixin, DeleteView):

    login\_url = '/auth/login'

    model = Accounts

    success\_message = "Account deleted successfully!"

    def get\_success\_url(self):

        return reverse("auth:manage\_customer")

class CreateAdminAccountView(LoginRequiredMixin, SuccessMessageMixin, CreateView):

    login\_url = '/auth/login'

    model = Accounts

    form\_class = AccountCreationForm

    template\_name = 'auth/add\_admin.html'

    success\_message = "Account created successfully!"

    def get\_success\_url(self):

        return reverse("auth:add\_admin")

        return Accounts.objects.filter(is\_staff=True).order\_by('-id')