ONLINE GROCEY STORE

MINI PROJECT REPORT

18CSC311J – WEB DESIGN AND DEVELOPMENT LABORATORY

(2018 Regulation)

III Year/ VI Semester

Academic Year: 2022 -2023

Submitted By

Kothoju Naresh [RA2111028010158]

Kolli Vineeth [RA2111028010179]

Under the guidance of

Dr. M. Sivakumar

Associate Professor

Department of Networking and Communications

DEPARTMENT OF NETWORKING AND COMMUNICATIONS

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Kancheepuram

MAY 2024

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603 203

BONAFIDE

Certified that this Mini project report titled "Online Grocery Store Application" for the course **18CSC311J – WEB DESIGN AND DEVELOPMENT LABORATORY** is the bonafide work of **Kothoju Naresh [RA2111028010158], Kolli Vineeth [RA2111028010179]** who undertook the task of completing the project within the allotted time.

Signature

Dr. M. Sivakumar

Course Faculty
Associate Professor
Department of NWC

Signature

Dr. Annapurani Panaiyappan K

Head of the Department
Professor
Department of NWC

ABSTRACT

This project focuses on the creation of an online grocery store web application using a comprehensive stack of web development technologies including HTML, CSS, JavaScript, PHP, and MySQL. The goal is to provide users with a convenient and efficient platform to purchase groceries online, offering a wide range of products categorized for easy navigation. Leveraging HTML for structure, CSS for styling, and JavaScript for interactivity, the front-end interface ensures a smooth and engaging user experience. Features like product filtering and search functionality enhance usability, while adhering to responsive design principles ensures accessibility across different devices.

On the back-end, PHP handles server-side logic and database interaction, while MySQL manages data storage and retrieval. This robust combination enables functionalities such as user authentication, product management, shopping cart handling, secure payment processing, order tracking, and user feedback management. The integration of modern web development practices and security measures ensures the reliability and scalability of the application, crucial for handling the complexities of an online grocery store.

Throughout the development process, emphasis is placed on user experience, security, and performance. Cross-browser compatibility and responsive design are prioritized to ensure seamless access and usability across various devices and screen sizes. By implementing this online grocery store web application, we showcase the practical application of HTML, CSS, JavaScript, PHP, and MySQL in delivering a fully functional and user-friendly e-commerce platform tailored to meet the evolving needs of online shoppers in the digital age.

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

In this chapter, we have given a description of the web application along with examples of existing web applications based on similar concepts and our project objectives.

1.1 About the Web Application

The online grocery store web application is designed to revolutionize the way users shop for groceries by offering a convenient and intuitive platform accessible from anywhere with an internet connection. Built on a foundation of HTML, CSS, JavaScript, PHP, and MySQL, this application provides a seamless experience for users to browse, select, and purchase groceries with ease. At the heart of the application is its user-friendly interface, crafted with HTML, CSS, and JavaScript to ensure a visually appealing and responsive design. Users can navigate through various product categories, utilize search and filtering options, and add items to their shopping cart effortlessly.

The interface is designed to prioritize usability, providing a smooth and intuitive experience for users of all levels of technical proficiency. Online grocery stores offer several benefits over traditional brick-and-mortar shops, such as reduced costs for the business, expanded reach to customers further away, and increased convenience for time-constrained or mobility-limited consumers. The COVID-19 pandemic has further accelerated the adoption of online grocery shopping as consumers sought safer alternatives to in-person shopping.

Behind the scenes, PHP and MySQL power the back-end functionalities of the application, handling tasks such as user authentication, product management, order processing, and data storage. Security measures are implemented to safeguard user information and ensure secure payment processing. Additionally, the application is optimized for performance and scalability, capable of handling high volumes of traffic and transactions without compromising speed or reliability. By leveraging cutting-edge web technologies, it offers a convenient, efficient, and secure platform for users to full fill their grocery shopping needs online, empowering them to save time and enjoy a seamless shopping experience from the comfort of their homes.

1.2 Existing Web Applications

Several existing web applications related to online grocery stores have made significant strides in the e-commerce landscape, catering to the growing demand for convenient and accessible grocery shopping experiences. Here are some existing web applications related to online grocery stores along with their features:

- **Instacart:** Instacart allows users to order groceries from local stores for delivery or pickup.
  - Wide range of participating stores.
  - Option to choose delivery windows.
  - Personalized shopping experience.
  - Integration with loyalty programs.

- **Amazon Fresh:** Amazon Fresh is Amazon's grocery delivery and pickup service.
  - Extensive selection of groceries and household items.
  - Delivery and pickup options.
  - Integration with Amazon Prime membership.
  - Fresh produce and perishable items available.

- **Walmart Grocery:** Walmart Grocery offers online grocery shopping with pickup and delivery options.
  - Large selection of groceries and household essentials.
  - Pickup and delivery options.
  - Savings with everyday low prices.
  - Scheduled pickups.
  - Easy reordering of frequently purchased items.

- **Fresh Direct:** Fresh Direct delivers fresh produce, groceries, and meal kits to customers in select areas.
  - Same-day delivery.
  - Fresh, high-quality produce.
  - Meal kits with pre-portioned ingredients.
  - Sustainable seafood options.
  - Integration with delivery pass for discounts.

1.3 Drawbacks of existing application

While existing online grocery store applications offer convenience and a wide range of features, they may also have certain drawbacks:

- **Delivery Fees:** Many online grocery store applications charge delivery fees, which can increase the overall cost of groceries, especially for smaller orders. Some users may find these fees prohibitive, leading them to seek alternative shopping methods.

- **Limited Availability:** Online grocery delivery services may not be available in all geographic areas, particularly in rural or remote locations. This limitation restricts access to convenient online shopping options for some consumers, forcing them to rely on traditional brick-and-mortar stores.

- **Product Availability and Substitutions:** Despite efforts to maintain accurate inventory, online grocery stores may experience shortages or discrepancies in product availability. This can result in substituted items or incomplete orders, leading to customer dissatisfaction and inconvenience.

- **Quality Control:** Ensuring the quality and freshness of perishable items like fruits, vegetables, and meats can be challenging in the online grocery delivery model. Some users may receive items that do not meet their expectations in terms of freshness or quality, leading to dissatisfaction with the service.

- **User Experience:** While online grocery store applications strive to provide a seamless user experience, some users may encounter technical glitches, slow loading times, or difficulties navigating the platform. These issues can detract from the overall shopping experience and frustrate users.

- **Environmental Impact:** The packaging and delivery processes associated with online grocery delivery services can contribute to environmental concerns, including excess packaging waste and carbon emissions from delivery vehicles. Some environmentally conscious consumers may be deterred from using these services due to their ecological footprint.

1.4 Project Objectives

- **Enhanced User Experience:** The first objective is to prioritize the user experience by creating an intuitive and seamless interface. This involves designing user-friendly navigation, clear product categorization, efficient search functionality, and a straightforward checkout process. By focusing on usability and accessibility, the aim is to ensure that users can easily find and purchase the groceries they need with minimal effort.

- **Scalability and Performance:** Another objective is to ensure the scalability and performance of the web application to accommodate a growing user base and handle increased traffic during peak periods. This involves optimizing the application's architecture, database structure, and server infrastructure to support high volumes of concurrent users and transactions

- **Security and Data Protection**: The third objective is to prioritize security and data protection to safeguard users' personal information, payment details, and transaction data.

- **Inventory Management:** Implement robust inventory management functionality to ensure accurate tracking of stock levels, product availability, and replenishment processes. This involves integrating real-time inventory updates, automated alerts for low stock levels, and seamless synchronization between the online platform and physical stores or warehouses.

- **Personalization and Recommendations:** Incorporate personalized shopping experiences by leveraging user data and browsing behaviour to offer tailored product recommendations, promotions, and discounts.

- **Analytics and Insights:** Integrate analytics and reporting capabilities to gather valuable insights into user behaviour, purchasing patterns, and performance metrics.

By focusing on these objectives, the online grocery store web application can deliver a reliable, user-friendly, and secure shopping experience that meets the needs and expectations of its users while also ensuring scalability and performance to support future growth and expansion.

CHAPTER 2

SYSTEM ARCHITECTURE

The chapter illustrates the web application's system architecture, accompanied by the use case and activity diagrams.

2.1 System Architecture

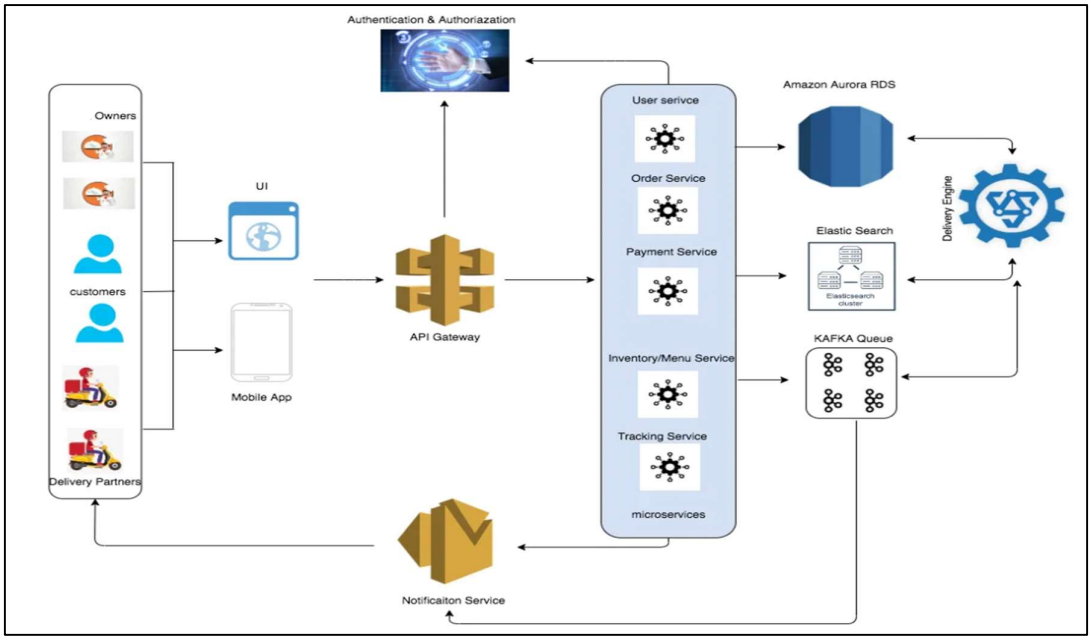

Fig 2.1: Architecture diagram for processing online grocery orders

- **Client Devices (Web Browser):** This represents the various client devices, such as laptops, desktops, smartphones, and tablets, used by users to access the online grocery store web application through web browsers.
- **Frontend:** The frontend consists of the user interface components of the web application developed using HTML, CSS, and JavaScript.
- **Web Server:** The web server hosts and serves the web application to client devices. It manages HTTP requests, processes dynamic content generation, and serves static files.
- **Backend:** The backend comprises server-side components responsible for handling business logic, processing requests from the frontend, and interacting with the database. It is typically implemented using server-side scripting languages such as PHP.
- **Database:** The database stores and manages the application's data, including product information, user profiles, order details, and inventory. In this architecture, MySQL is used as the relational database management system (RDBMS) for data storage and retrieval.

2.2 Use Case Diagram

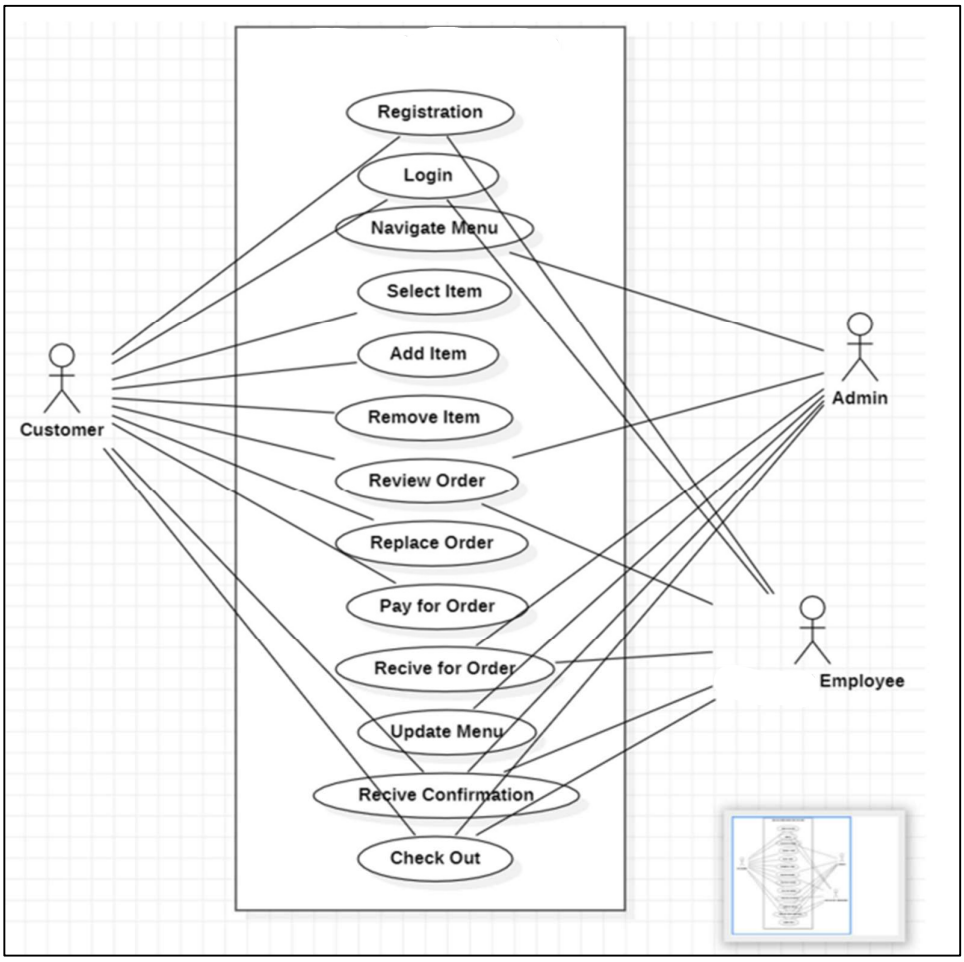

Fig 2.2: Usecase diagram for online grocery store

The use cases in the diagram represents the main processes in an Online Grocery store. Then they will be broken down into more specific use cases depending on the included processes of the main use case. Each of these use cases explains how the system handles the actions or scenarios requested by the user. The UML Use Case Diagram is a design used as one of the Methodology on online grocery store development.

Use cases: Registration, Login, Navigate Menu, Select Item, Add Item, Remove Item, Review Order, Replace Order, Pay for Order, Recive for Order, Update Menu, Check Out

Actors: Customers, Admin, Employee

These use cases encompass the core actions that customers can perform while using the online grocery store, facilitating a seamless and intuitive shopping experience tailored to their needs and preferences.

2.3 Activity Diagram

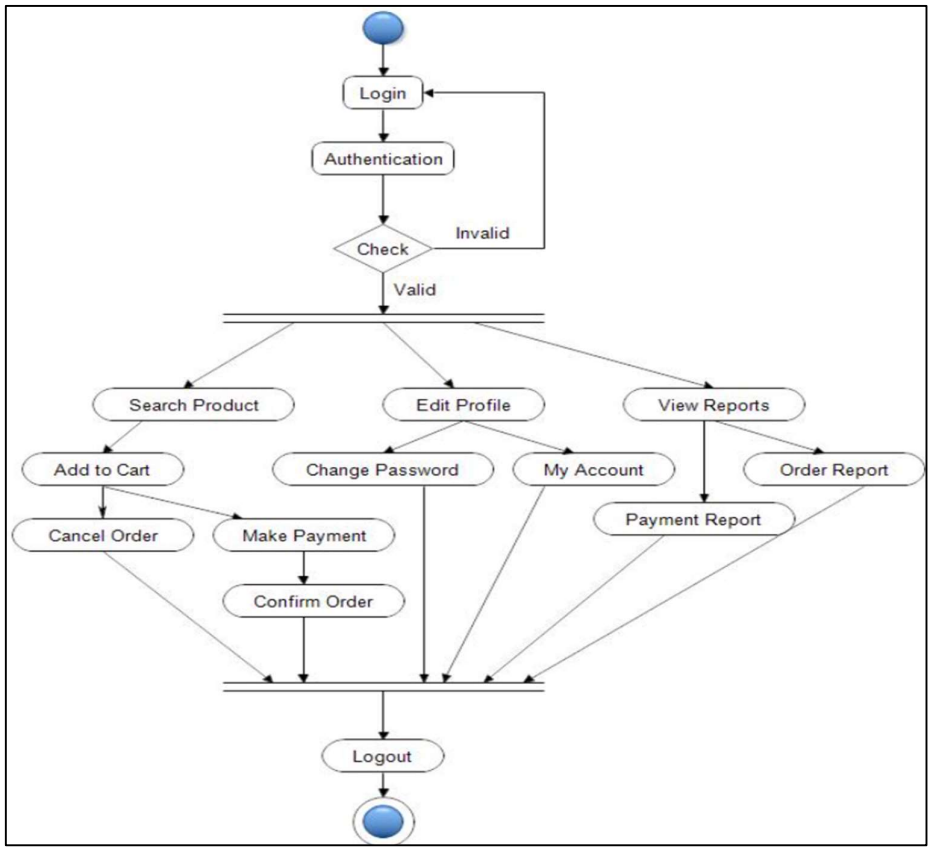

Fig 2.3: Activity diagram for online grocery store

The UML activity Diagram is used to show the interaction of the user and the system. By creating it, you'll be able to see the flaws of the system and you may avoid it once you apply it to the project development. So it is important to have your diagrams designed first before jumping into its development. Now to create this kind of diagram, you have to determine the processes involve, the users and finalize the behaviour of your proposed system. Symbols and Notations Activity Is used to illustrate a set of actions. It shows the non-interruptible action of objects.

An object flow arrow from an action to an object means that the action creates or influences the object. An object flow arrow from an object to an action indicates that the action state uses the object. Decisions and Branching A diamond represents a decision with alternate paths. When an activity requires a decision prior to moving on to the next activity, add a diamond between the two activities.

CHAPTER 3

WEB SITE FRONT END IMPLEMENTATION

In this chapter, we have mentioned the tools that were used for the front-end implementation along with the codes in the languages used.

3.1 Tools used

For the frontend development of an online grocery store web application, various tools and technologies can be utilized to create a visually appealing and user-friendly interface.

- **HTML (Hypertext Markup Language):** HTML is the standard markup language used to create the structure and content of web pages. It provides the foundation for building the layout and organizing the elements of the online grocery store's interface.
- **CSS (Cascading Style Sheets):** CSS is used to style and format the visual presentation of HTML elements, including colors, fonts, layout, and overall aesthetics. It enables designers to customize the appearance of the online grocery store to create a cohesive and visually appealing user experience.
- **JavaScript:** JavaScript is a dynamic scripting language that adds interactivity and behavior to web pages. It is commonly used to implement features such as product filtering, search functionality, image sliders, enhancing the user experience of the online grocery store.
- **Bootstrap:** Bootstrap is a popular front-end framework that provides pre-designed templates, components, and CSS stylesheets for building responsive and mobile-first web applications. It simplifies the process of creating responsive layouts,speeding up the development process and ensuring consistency across different devices and screen sizes.
- **JQuery:** jQuery is a fast, small, and feature-rich JavaScript library that simplifies HTML document traversing, event handling, animating, and Ajax interactions. It is commonly used alongside JavaScript to streamline DOM manipulation and simplify complex tasks, making it easier to develop interactive and dynamic features for the online grocery store's frontend.
- **Ajax (Asynchronous JavaScript and XML):** Ajax is a technique used to make asynchronous HTTP requests from the client-side to the server-side without reloading the entire web page. It enables developers to fetch data from the server and update specific parts of the web page dynamically, enhancing the user experience by providing seamless and responsive interactions.

3.2 Web Page Design

index.php (User)

```
<?php
session_start();
require 'dbcon.php';
$result = $conn->query('SELECT * FROM `product` LIMIT 8');
?>
<!DOCTYPE html>
<html>
<head>
<title>Grocery Store </title>
<body>
<?php include 'header.php'?>
        <div class="w3l_banner_nav_right">
            <section class="slider">
                <div class="flexslider">
                    <ul class="slides">
                        <li>
                            <div class="w3l_banner_nav_right_banner">
                                <h3>Make your <span>food</span> with Spicy.</h3>
                                <div class="more">
                                    <a href="products.php" class="button--saqui button--round-l
button--text-thick" data-text="Shop now">Shop now</a>
                                </div>
                            </div>
                        </li>
                        <li>
                            <div class="w3l_banner_nav_right_banner1">
                                <h3>Make your <span>food</span> with Spicy.</h3>
                                <div class="more">
                                    <a href="products.php" class="button--saqui button--round-l
button--text-thick" data-text="Shop now">Shop now</a>
                                </div>
                            </div>
                        </li>
                        <li>
                            <div class="w3l_banner_nav_right_banner">
                                <h3>upto <i>50%</i> off.</h3>
                                <div class="more">
                                    <a href="products.php" class="button--saqui button--round-l
button--text-thick" data-text="Shop now">Shop now</a>
                                </div>
                            </div>
                        </li>
                    </ul>
                </div>
            </section>
```

```
            <!-- flexSlider -->
                        <link rel="stylesheet" href="css/flexslider.css" type="text/css"
media="screen" property="" />
                <script defer src="js/jquery.flexslider.js"></script>
                <script type="text/javascript">
                $(window).load(function(){
                  $('.flexslider').flexslider({
                    animation: "slide",
                    start: function(slider){
                      $('body').removeClass('loading');
                    }
                  });
                });
              </script>
</div>

<!-- top-brands -->
<div class="top-brands">
   <div class="container">
      <h3>Hot Offers</h3>
      <div class="agile_top_brands_grids">
         <?php
         if ($result->num_rows) {
            while ($product = $result->fetch_assoc()) {
               $pid = $product['pid'];
               $name = $product['name'];
               $weight = trim($product['weight'], '()');
               $pic = $product['pic'];
               $price = $product['price'];
               $discount = $product['discount'];
               $discount_money = $price * ($product['discount'] / 100);
               $new_price = $discount == 0
                  ? $price
                  : $product['price'] * (1 - ($product['discount'] / 100)); ?>
         <div class="col-md-3 top_brand_left" style="margin-bottom:15px">
            <div class="hover14 column">
               <div class="agile_top_brand_left_grid">
                  <div class="tag">
                     <img src="images/tag.png" alt="" class="img-responsive">
                  </div>
                  <div class="agile_top_brand_left_grid1">
                     <figure>
                        <div class="snipcart-item block" >
                           <div class="snipcart-thumb">
                              <a href="single.php?id=<?php echo $pid; ?>">
                               <img title="<?php echo $name; ?>" alt="<?php echo $name;
?>" src="<?php echo $pic; ?>" width="140">
                              </a>
                              <p>
                                 <?php echo $name.($weight ? " ($weight)" : ''); ?>
```

```
                </ul>
            </div>
        </div>
        <div class="col-md-9 w3l_fresh_vegetables_grid_right">
            <div class="col-md-4 w3l_fresh_vegetables_grid">
                <div class="w3l_fresh_vegetables_grid1">
                    <img src="images/8.jpg" alt=" " class="img-responsive" />
                </div>
            </div>
            <div class="col-md-4 w3l_fresh_vegetables_grid">
                <div class="w3l_fresh_vegetables_grid1">
                    <div class="w3l_fresh_vegetables_grid1_rel">
                        <img src="images/7.jpg" alt=" " class="img-responsive" />
                        <div class="w3l_fresh_vegetables_grid1_rel_pos">
                            <div class="more m1">
                                <a href="products.php" class="button--saqui button--round-l
button--text-thick" data-text="Shop now">Shop now</a>
                            </div>
                        </div>
                    </div>
                </div>
                <div class="w3l_fresh_vegetables_grid1_bottom">
                    <img src="images/10.jpg" alt=" " class="img-responsive" />
                    <div class="w3l_fresh_vegetables_grid1_bottom_pos">
                        <h5>Special Offers</h5>
                    </div>
                </div>
            </div>
            <div class="col-md-4 w3l_fresh_vegetables_grid">
                <div class="w3l_fresh_vegetables_grid1">
                    <img src="images/9.jpg" alt=" " class="img-responsive" />
                </div>
                <div class="w3l_fresh_vegetables_grid1_bottom">
                    <img src="images/11.jpg" alt=" " class="img-responsive" />
                </div>
            </div>
            <div class="clearfix"> </div>
            <div class="agileinfo_move_text">
                <div class="agileinfo_marquee">
                    <h4>get <span class="blink_me">25% off</span> on first order and
also get gift voucher</h4>
                </div>
                <div class="agileinfo_breaking_news">
                    <span> </span>
                </div>
</div>
<?php include 'footer.php'?>
</body>
</html>
```

index.php (Admin)

```
<?php
session_start();
if (!isset($_SESSION['ADMIN_ID']) || empty($_SESSION['ADMIN_ID'])) {
   header('Location: login.php');
   exit;
}
require '../dbcon.php';

$result = $conn->query('SELECT `cid` FROM `category`');
$category = $result->num_rows;

$result = $conn->query('SELECT `pid` FROM `product`');
$product = $result->num_rows;

$result = $conn->query('SELECT `uid` FROM `user`');
$user = $result->num_rows;

$result = $conn->query('SELECT `fid` FROM `feedback`');
$feedback = $result->num_rows;
$result = $conn->query('SELECT `oid` FROM `ord`');
$ord = $result->num_rows;
$result = $conn->query('SELECT `payid` FROM `payment`');
$payment = $result->num_rows;
?>
<!DOCTYPE html>
<html lang="en">
<head>
<?php
   $title = 'Dashboard | Admin';
   require 'include/head.php';
?>
      <!-- End of Topbar -->
      <!-- Begin Page Content -->
      <div class="container-fluid">
        <!-- Page Heading -->
        <div class="d-sm-flex align-items-center justify-content-between mb-4">
          <h1 class="h3 mb-0 text-gray-800">Dashboard</h1>
              <!--<a href="#" class="d-none d-sm-inline-block btn btn-sm btn-primary
shadow-sm"><i class="fas fa-download fa-sm text-white-50"></i> Generate
Report</a>-->
        </div>
        <!-- Content Row -->
        <div class="row">
          <div class="col-xl-3 col-md-6 mb-4">
            <div class="card border-left-primary shadow h-100 py-2">
              <div class="card-body">
                <div class="row no-gutters align-items-center">
                  <div class="col mr-2">
```

```
                <div class="text-xs font-weight-bold text-warning text-uppercase mb-1">Total Users</div>
                 <div class="h5 mb-0 font-weight-bold text-gray-800"><?=$user?></div>
                </div>
                <div class="col-auto">
                  <i class="fas fa-comments fa-2x text-gray-300"></i>
                </div>
               </div>
          <div class="col-xl-3 col-md-6 mb-4">
            <div class="card border-left-danger shadow h-100 py-2">
              <div class="card-body">
               <div class="row no-gutters align-items-center">
                <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-warning text-uppercase mb-1">Total Products</div>
                                    <div class="h5 mb-0 font-weight-bold text-gray-800"><?=$product?></div>
                    <div class="text-xs font-weight-bold text-warning text-uppercase mb-1">Total Orders</div>
                     <div class="h5 mb-0 font-weight-bold text-gray-800"><?=$ord?></div>
                </div>
                <div class="col-auto">
                  <i class="fas fa-comments fa-2x text-gray-300"></i>
          <div class="col-xl-3 col-md-6 mb-4">
            <div class="card border-left-danger shadow h-100 py-2">
              <div class="card-body">
               <div class="row no-gutters align-items-center">
                <div class="col mr-2">
                    <div class="text-xs font-weight-bold text-warning text-uppercase mb-1">Total Payments</div>
                                    <div class="h5 mb-0 font-weight-bold text-gray-800"><?=$payment?></div>
                </div>
                <div class="col-auto">
                  <i class="fas fa-comments fa-2x text-gray-300"></i
        </div>
        <!-- /.container-fluid -->
           <span>Copyright &copy; 2024 Grocery Store. All rights reserved | Design and Developed by Naresh & Vineeth</span>
         </div>
        </div>
      </footer>
<?php
   require 'include/javascript.php';
?>
  <!-- Page level custom scripts -->
  <script src="js/demo/chart-area-demo.js"></script>
  <script src="js/demo/chart-pie-demo.js"></script>
</body>
</html>
```

style.css

```
ul.resp-tabs-list, p {
    margin: 0px;
    padding: 0px;
}

.resp-tabs-list li {
    font-weight: 400;
    font-size: 20px;
    letter-spacing: 2px;
    display: inline-block;
    padding: 13px 15px;
    margin: 0 4px 0 0;
    list-style: none;
    cursor: pointer;
    float: left;
}

.resp-tabs-container {
    padding: 0px;
    background-color: #fff;
    clear: left;
}

h2.resp-accordion {
    cursor: pointer;
    padding: 5px;
    display: none;
}

.resp-tab-content {
    display: none;
    padding: 25px 60px;
}

.resp-tab-active {
    border: 1px solid #212121 !important;
    border-bottom: none;
    margin-bottom: -1px !important;
    padding: 12px 14px 14px 14px !important;
    border-bottom: 0px #fff solid !important;
}
.resp-vtabs .resp-tabs-list li i {
    color: #000000;
    margin-right: 20px;
    padding: 15px 10px;
    width:50px;
    height: 50px;
    text-align: center;
```

```
h2.resp-tab-title:last-child {
   border-bottom: 12px solid #c1c1c1 !important;
   background: blue;
}
/*----------Vertical tabs----------*/
.agileits-tab_nav {
   float: left;
   width: 30%;
   margin-top: 0!important;
   min-height: 650px;
}
ul.resp-tabs-list.hor_1 {
   margin-top: 0!important;
}
a.w3ls-ads {
   text-decoration: none;
   color: #0099e5;
   font-size: 15px;
   margin: 25px 0 0px 0;
   display: block;
   text-align: center;
}
a.w3ls-ads:hover {
   color:#ff4c4c;
}
.resp-vtabs .resp-tabs-container {
   padding: 0px;
   background-color: rgba(243, 243, 243, 0.25);
   border: none;
   width: 70%;
   border: 1px solid transparent !important;
   border-radius: 0;
   clear: none;
   min-height: 650px;
}

.resp-vtabs .resp-tab-content {
   word-wrap: break-word;
}
/*--
li.resp-tab-item.hor_1.resp-tab-active:after {
   content: '';
   position: absolute;
   border-left: 1px solid #5AB1D0;
   border-right: 0px solid #5AB1D0;
   border-bottom: 1px solid #FFFFFF;
   transform: rotate(134deg);
   border-top: 1px solid #5AB1D0;
   padding: 0 33px 33px 0px;
}
```

bootstrap.css

```
html {
  font-family: sans-serif;
  -webkit-text-size-adjust: 100%;
      -ms-text-size-adjust: 100%;
}
body {
  margin: 0;
}
article,
aside,
details,
figcaption,
figure,
footer,
header,
hgroup,
main,
menu,
nav,
section,
summary {
  display: block;
}
audio,
canvas,
progress,
video {
  display: inline-block;
  vertical-align: baseline;
}
audio:not([controls]) {
  display: none;
  height: 0;
}
[hidden],
template {
  display: none;
}
a {
  background-color: transparent;
}
a:active,
a:hover {
  outline: 0;
}
abbr[title] {
  border-bottom: 1px dotted;
}
```

```
b,
strong {
  font-weight: bold;
}
dfn {
  font-style: italic;
}
h1 {
  margin: .67em 0;
  font-size: 2em;
}
mark {
  color: #000;
  background: #ff0;
}
small {
  font-size: 80%;
}
sub,
sup {
  position: relative;
  font-size: 75%;
  line-height: 0;
  vertical-align: baseline;
}
sup {
  top: -.5em;
}
sub {
  bottom: -.25em;
}
img {
  border: 0;
}
svg:not(:root) {
  overflow: hidden;
}
figure {
  margin: 1em 40px;
}
hr {
  height: 0;
  -webkit-box-sizing: content-box;
     -moz-box-sizing: content-box;
          box-sizing: content-box;
}
pre {
  overflow: auto;
}
```

minicart.js

```
// Easy Responsive Tabs Plugin
// Author: Samson.Onna <Email : samson3d@gmail.com>
(function ($) {
    $.fn.extend({
        easyResponsiveTabs: function (options) {
            //Set the default values, use comma to separate the settings, example:
            var defaults = {
                type: 'default', //default, vertical, accordion;
                width: 'auto',
                fit: true,
                closed: false,
                tabidentify: '',
                activetab_bg: 'white',
                inactive_bg: '#F5F5F5',
                active_border_color: '#c1c1c1',
                active_content_border_color: '#c1c1c1',
                activate: function () {
                }
            }
            //Variables
            var options = $.extend(defaults, options);
             var opt = options, jtype = opt.type, jfit = opt.fit, jwidth = opt.width, vtabs =
'vertical', accord = 'accordion';
            var hash = window.location.hash;
            var historyApi = !!(window.history && history.replaceState);

            //Events
            $(this).bind('tabactivate', function (e, currentTab) {
                if (typeof options.activate === 'function') {
                    options.activate.call(currentTab, e)
                }
            });

            //Main function
            this.each(function () {
                var $respTabs = $(this);
                var $respTabsList = $respTabs.find('ul.resp-tabs-list.' + options.tabidentify);
                var respTabsId = $respTabs.attr('id');
                $respTabs.find('ul.resp-tabs-list.' + options.tabidentify + ' li').addClass('resp-
tab-item').addClass(options.tabidentify);
                $respTabs.css({
                    'display': 'block',
                    'width': jwidth
                });

                if (options.type == 'vertical')
                    $respTabsList.css('margin-top', '3px');
```

```
            $respTabs.find('.resp-tabs-container.' + options.tabidentify).css('border-color',
options.active_content_border_color);
                        $respTabs.find('.resp-tabs-container.' + options.tabidentify + ' >
div').addClass('resp-tab-content').addClass(options.tabidentify);
            jtab_options();
            //Properties Function
            function jtab_options() {
               if (jtype == vtabs) {
                  $respTabs.addClass('resp-vtabs').addClass(options.tabidentify);
               }
               if (jfit == true) {
                  $respTabs.css({ width: '100%', margin: '0px' });
               }
               if (jtype == accord) {
                                                                  $respTabs.addClass('resp-easy-
accordion').addClass(options.tabidentify);
                  $respTabs.find('.resp-tabs-list').css('display', 'none');
               }
            }

            //Assigning the h2 markup to accordion title
            var $tabItemh2;
                  $respTabs.find('.resp-tab-content.' + options.tabidentify).before("<h2
class='resp-accordion " + options.tabidentify + "' role='tab'><span class='resp-
arrow'></span></h2>");

            $respTabs.find('.resp-tab-content.' + options.tabidentify).prev("h2").css({
               'background-color': options.inactive_bg,
               'border-color': options.active_border_color
            });

            var itemCount = 0;
            $respTabs.find('.resp-accordion').each(function () {
               $tabItemh2 = $(this);
               var $tabItem = $respTabs.find('.resp-tab-item:eq(' + itemCount + ')');
               var $accItem = $respTabs.find('.resp-accordion:eq(' + itemCount + ')');
               $accItem.append($tabItem.html());
               $accItem.data($tabItem.data());
                     $tabItemh2.attr('aria-controls', options.tabidentify + '_tab_item-' +
(itemCount));
               itemCount++;
            });
            //Window resize function
            $(window).resize(function () {
               $respTabs.find('.resp-accordion-closed').removeAttr('style');
            });
         });
      }
   });
})(jQuery);
```

3.3 Screenshots

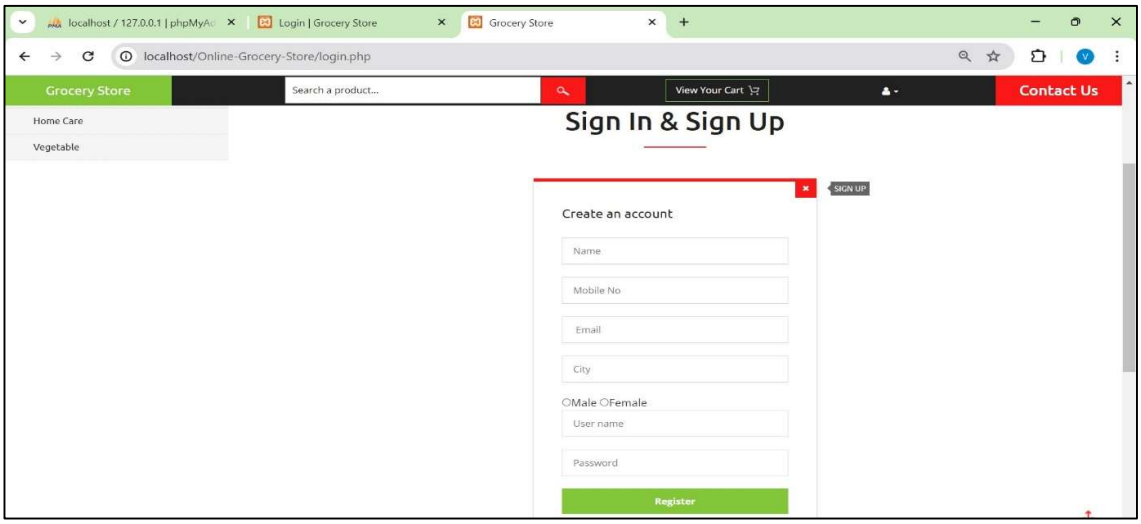

Fig 3.1: User signup page

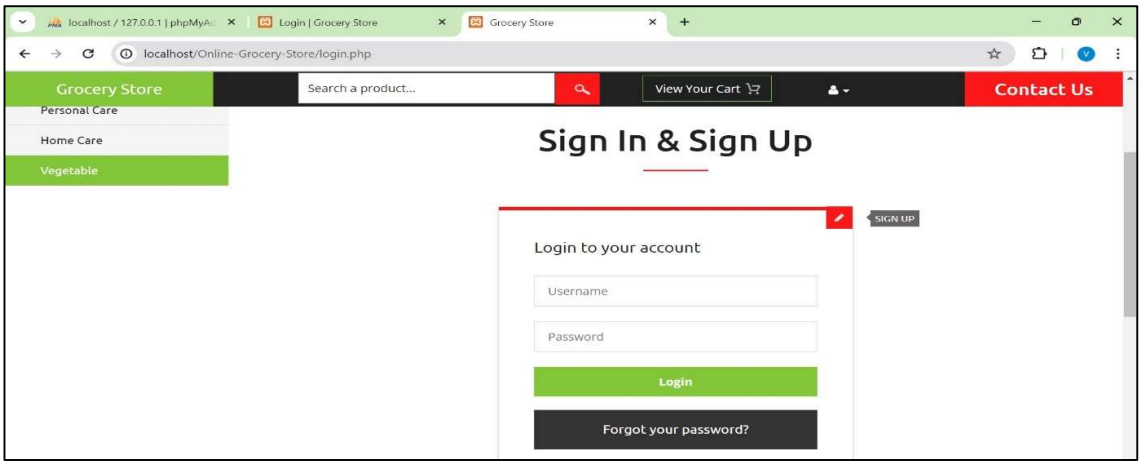

Fig 3.2: User signin page

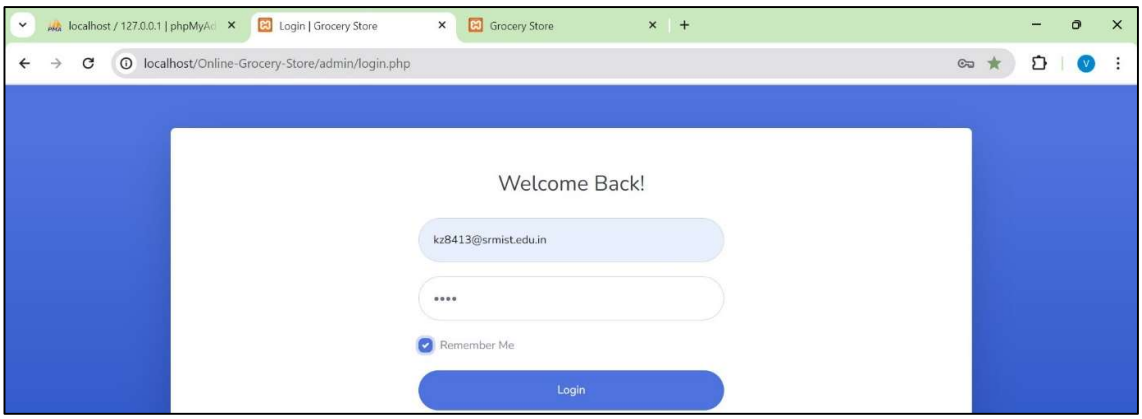

Fig 3.3 Admin signin page

CHAPTER 4

WEB SITE BACK-END IMPLEMENTATION

In this chapter, we have mentioned the tools that were used for the back-end implementation along with the codes in the languages used.

4.1 Tools used

In our online grocery store web application, PHP tools serve as the backbone of your backend operations, facilitating the dynamic processing of data and seamless interaction between your application and its database. Let's delve into the key PHP tool commonly used in such scenarios:

- **PHP (Hypertext Preprocessor):** PHP is a widely-used open-source scripting language that is particularly suited for web development and can be embedded into HTML. In your online grocery store application, PHP serves as the primary server-side language, handling dynamic content generation, data processing, and interaction with the database. In the realm of backend development for your online grocery store web application, PHP emerges as a cornerstone technology. PHP, renowned for its server-side scripting capabilities, empowers your backend operations with dynamic content generation, seamless database interactions, and robust user authentication mechanisms.

  By harnessing PHP frameworks like Laravel or Symfony, you can construct a resilient and scalable backend architecture tailored precisely to your grocery store's requirements. With PHP, managing inventory, processing orders, securing payments, and ensuring smooth communication between the frontend and database becomes not only feasible but efficient. Its flexibility and extensive community support render PHP an indispensable tool for powering the functionality of your online grocery store, guaranteeing seamless user experiences and streamlined business operations.

4.2 Code for Back-end

Viewcart.php

```
<?php
$products = $cart = [];

foreach ($_POST as $key => $val) {
   $array = explode('_', $key); // discount_amount_1 => Array([0] => discount, [1] =>
amount, [2] => 1)

   if (count($array) > 1) { // exclude keys without _
      $i = array_pop($array); // get and remove last member of array => Array([0] =>
discount, [1] => amount)
   } else {
      $i = $array[0];
   }

     $key = implode('_', $array); // Array([0] => discount, [1] => amount) =>
discount_amount

   if (is_numeric($i)) {
      $products[$i][$key] = $val;
   } else {
      $cart[$key] = $val;
   }
}

require 'dbcon.php';

?>

<!DOCTYPE html>
<html>
<head>
   <title>My Cart | Grocery Store</title>

<?php
   include 'header.php'?>
      <form action="checkout.php" method="post">
         <div class="w3l_banner_nav_right">
            <!-- about -->
            <div class="privacy about">
               <h3>My <span>Cart</span></h3>
               <div class="checkout-right">
                        <h4>Your shopping cart contains: <span><?=count($products)?>
Products</span></h4>
                  <table class="timetable_sub">
```

```
                                                                    <span
id="amount_<?=$pid?>"><?=$product['amount']?></span>
                        </td>
                                    <input type="hidden" name="amount_<?=$pid?>"
value="<?=$product['amount']?>">

                        <td class="invert">
                            <div class="quantity">
                              <div class="quantity-select">
                                 <div class="entry value-minus" data-
                                     <input type="hidden" name="quantity_<?=$pid?>"
value="<?=$product['quantity']?>">

                        <td class="invert">
                           <span id="subtotal_<?=$pid?>"><?=$subtotal?></span>
                        </td>
                                     <input type="hidden" name="subtotal_<?=$pid?>"
value="<?=$subtotal?>">

                        <td class="invert">
                           <div class="rem" id="<?$i?>">
                              <div class="close1"></div>
                           </div>
                        </td>
                     </tr>
                     <?php
                     }
                     ?>
                  </tbody>
                </table>
              </div>
              <div class="checkout-left">
                 <div class="col-md-4 checkout-left-basket">
                    <ul>
                          <li style="font-size: 20px">Delivery Charges <i>-</i> <span
class="text-primary">Free</span></li>
                        <li><hr></li>
                           <li style="font-weight: bolder;font-size: 20px">Total  <span
id="total"><?=$total?></span> <span>₹</span></li>
                        <input type="hidden" name="total" value="<?=$total?>">
                     </ul>
                     <div class="row">
                     </div>
                  </div>
                  <div class="col-md-8 address_form_agile">
                        <section class="creditly-wrapper wthree, w3_agileits_wrapper"
style="margin-top: 35px">
                        <div class="information-wrapper">
                                    <button class="submit check_out btn-block">Place
Order</button>
```

```
                    </div>
                </section>
            </div>
            <div class="clearfix"></div>
        <!-- //about -->
    </div>
</form>
<div class="clearfix"></div>
</div>
<!-- //banner -->
<?php include 'footer.php'?>
$('.value-minus').on('click', function() {
    id = $(this).attr('data-id');
    var divUpd = $(this).parent().find('.value');
    quantity1 = parseInt(divUpd.text(), 10)
    quantity2 = quantity1 - 1;
    if(quantity2 < 1) { quantity2 = 1; }
    divUpd.text(quantity2);
    $('[name=quantity_'+id+']').val(quantity2);
    amount = Number($('#amount_'+id).text());
    subtotal = quantity2 * amount;
    $('[name=subtotal_'+id+']').val(subtotal);
    $('#subtotal_'+id).text(subtotal);
    total1 = Number($('#total').text());
    total2 = total1 + (amount * (quantity2 - quantity1));
    $('#total').text(total2);
    $('[name=total]').val(total2);
});

$(document).ready(function(c) {
    $('.rem').on('click', function(c) {
        id = $(this).attr("id");
        $('.rem'+id).fadeOut('slow', function(c){
            $('.rem'+id).remove();
        });

$(document).ready(function(c) {
    $('.close2').on('click', function(c){
        $('.rem2').fadeOut('slow', function(c){
            $('.rem2').remove();
        });
$(document).ready(function(c) {
    $('.close3').on('click', function(c){
        $('.rem3').fadeOut('slow', function(c){
            $('.rem3').remove();
        });
    });
</script>
</body>
</html>
```

CHAPTER 5

DATABASE DESIGN

In this chapter, we have mentioned the database that was implemented for the project along with the schema structure and tables that were used.

5.1 Database used

- **SQL Database Management:** In the backend of your online grocery store web application, SQL databases play a pivotal role in storing and managing crucial data. SQL, or Structured Query Language, is the standard language for interacting with relational databases, providing a robust framework for organizing, querying, and manipulating data efficiently. By employing relational database management systems (RDBMS), you ensure data integrity, scalability, and reliability, essential for the seamless operation of your online grocery store.

5.2 Database schema and Tables

grocery.sql

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

--
-- Database: `grocery`
--
CREATE DATABASE IF NOT EXISTS `grocery` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
USE `grocery`;

-- --------------------------------------------------------

--
-- Table structure for table `admin`
--

DROP TABLE IF EXISTS `admin`;
CREATE TABLE `admin` (
  `aid` int(10) NOT NULL,
  `name` varchar(50) NOT NULL,
  `email` varchar(50) NOT NULL,
  `password` int(10) NOT NULL
```

```
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
INSERT INTO `admin` (`aid`, `name`, `email`, `password`) VALUES
(1, 'himani', 'himaniaasoda1999@gmail.com', 8814);

-- --------------------------------------------------------

--
-- Table structure for table `category`
--

DROP TABLE IF EXISTS `category`;
CREATE TABLE `category` (
  `cid` int(10) NOT NULL,
  `name` varchar(50) DEFAULT NULL,
  `parent_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `category`
--

INSERT INTO `category` (`cid`, `name`, `parent_id`) VALUES
(1, 'grocery', 0),
(2, 'Personal care', 0),
(3, 'home care', 0),
(4, 'Vegetable', 0),
(5, 'masala and spice', 1),
(6, 'oil rice', 1),
(7, 'pluses', 1),
(8, 'Cereal', 1);

--
-- Table structure for table `feedback`
DROP TABLE IF EXISTS `feedback`;
CREATE TABLE `feedback` (
  `fid` int(10) NOT NULL,
  `name` varchar(50) DEFAULT NULL,
  `mobile` varchar(12) DEFAULT NULL,
  `msg` text DEFAULT NULL,
  `uid` int(10) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
-- Dumping data for table `feedback`
INSERT INTO `feedback` (`fid`, `name`, `mobile`, `msg`, `uid`) VALUES
(1, 'Barak Obama', '9876543210', 'I am very impressed with this amazing website.',
NULL);

-- --------------------------------------------------------

--
-- Table structure for table `ord`
```

```
--

DROP TABLE IF EXISTS `ord`;
CREATE TABLE `ord` (
  `oid` int(10) NOT NULL,
  `uid` int(11) DEFAULT NULL,
  `total` int(11) DEFAULT NULL,
  `date` datetime DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
-- Dumping data for table `ord`

INSERT INTO `ord` (`oid`, `uid`, `total`, `date`) VALUES
(1, 1, 240, '2020-10-12 17:31:03');
--
-- Table structure for table `order_items`
--

DROP TABLE IF EXISTS `order_items`;
CREATE TABLE `order_items` (
  `item_id` int(11) NOT NULL,
  `oid` int(11) NOT NULL,
  `pid` int(11) NOT NULL,
  `quantity` int(11) DEFAULT NULL,
  `amount` int(11) DEFAULT NULL,
  `subtotal` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `order_items`
--

INSERT INTO `order_items` (`item_id`, `oid`, `pid`, `quantity`, `amount`, `subtotal`)
VALUES
(1, 1, 1, 2, 50, 100),
(2, 1, 2, 10, 14, 140);
DROP TABLE IF EXISTS `product`;
CREATE TABLE `product` (
  `pid` int(10) NOT NULL,
  `name` varchar(50) DEFAULT NULL,
  `price` float DEFAULT NULL,
  `discount` int(10) DEFAULT NULL,
  `weight` varchar(20) DEFAULT NULL,
  `pic` varchar(100) DEFAULT NULL,
  `cid` int(10) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

-- Table structure for table `user`
DROP TABLE IF EXISTS `user`;
CREATE TABLE `user` (
  `uid` int(10) NOT NULL,
```

```
  `name` varchar(50) NOT NULL,
  `mobile` bigint(10) NOT NULL,
  `address1` varchar(50) NOT NULL,
  `gender` enum('male','female') NOT NULL,
  `username` varchar(20) NOT NULL,
  `password` varchar(10) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `user`
--

INSERT INTO `user` (`uid`, `name`, `mobile`, `address1`, `gender`, `username`,
`password`) VALUES
(1, 'himani', 9106682997, 'visnager, VILLAGE=ASODA', 'female', 'himani', 'hd6034'),
(2, 'divya', 9979331605, 'visnager, VILLAGE=ASODA', 'female', 'daxrp', 'dk7845'),
(3, 'kishan', 9714304713, 'visnager', 'male', 'kishan rp', 'kr9856'),
(4, 'rajubhai', 9979331605, 'asoda', 'male', 'daxrp', 'rp1236'),
(5, 'sonali', 1236547898, 'visnager,', 'female', 'sonali', 'sp1236');

--
-- Indexes for dumped tables
--

--
-- Indexes for table `admin`
--
ALTER TABLE `admin`
  ADD PRIMARY KEY (`aid`);

--
-- Indexes for table `category`
--
ALTER TABLE `category`
  ADD PRIMARY KEY (`cid`);

--
-- Indexes for table `feedback`
--
ALTER TABLE `feedback`
  ADD PRIMARY KEY (`fid`),
  ADD KEY `feedback_ibfk_1` (`uid`);

--
-- Indexes for table `ord`
--
ALTER TABLE `ord`
  ADD PRIMARY KEY (`oid`),
  ADD KEY `user_id` (`uid`);
```

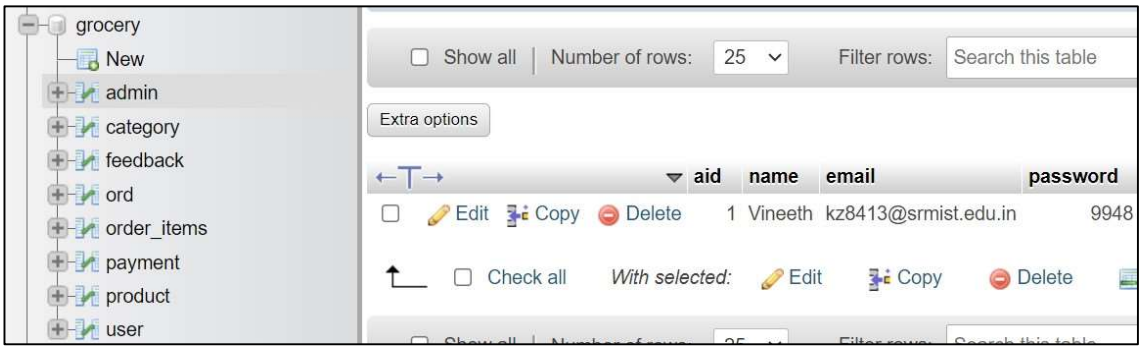

Fig 5.1: Admin database table

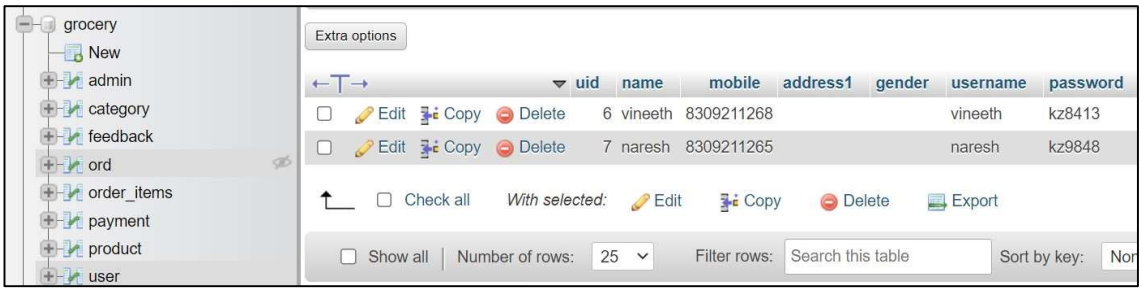

Fig 5.2: User database table

Fig 5.3: Product database table

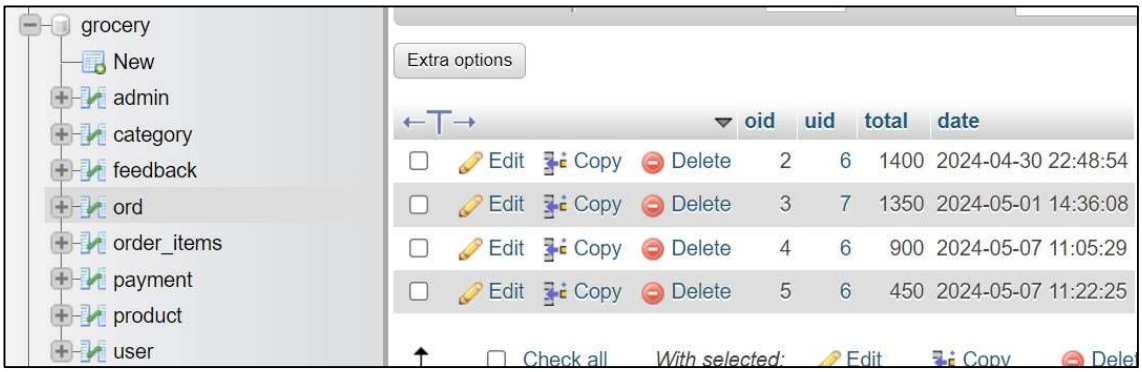

Fig 5.4: Order database table

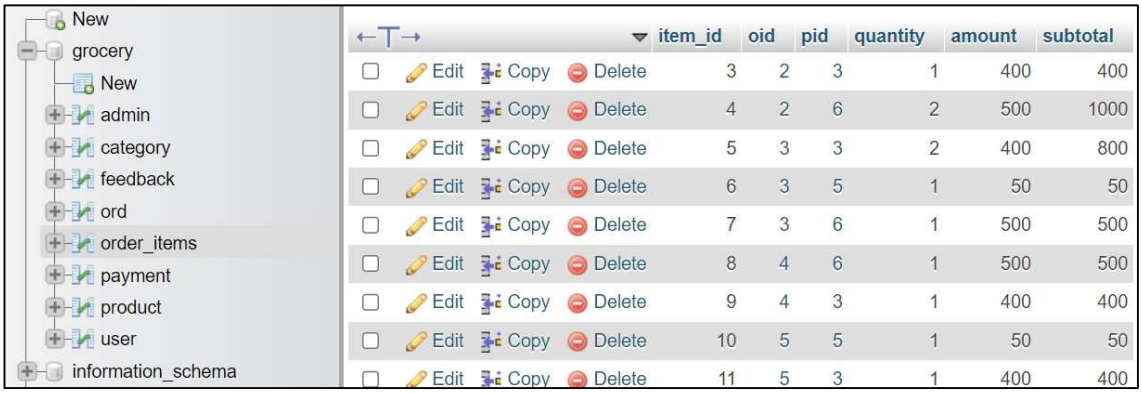

Fig 5.5: Ordered items database table

Fig 5.6: Payment database table

5.3 Database Connectivity

Efficient database connectivity is essential for ensuring smooth communication between your online grocery store web application and the underlying SQL database. Through PHP, you can establish a reliable connection to the database server, allowing seamless retrieval, manipulation, and storage of data. PHP offers built-in functions and libraries like PDO (PHP Data Objects) and MySQLi (MySQL Improved) for interacting with SQL databases, enabling secure and efficient database operations. By establishing persistent connections, ensuring consistent and high-performance database connectivity for your online grocery store.

Furthermore, incorporating connection pooling techniques helps manage database connections efficiently, especially in high-traffic environments, by reusing existing connections instead of creating new ones for each user request.By prioritizing scalability, reliability, and performance in database connectivity, you can ensure that your online grocery store web application delivers a seamless user experience, with fast response times and reliable data access, even during peak usage periods.

CHAPTER 6

SCREENSHOTS

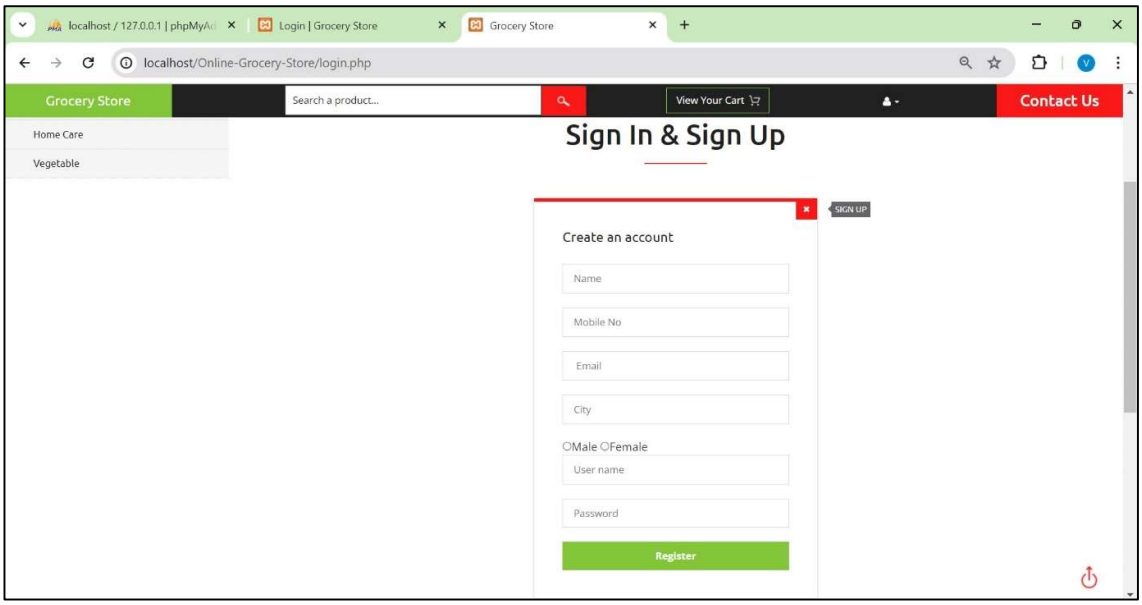

Fig 6.1: User Signup Page

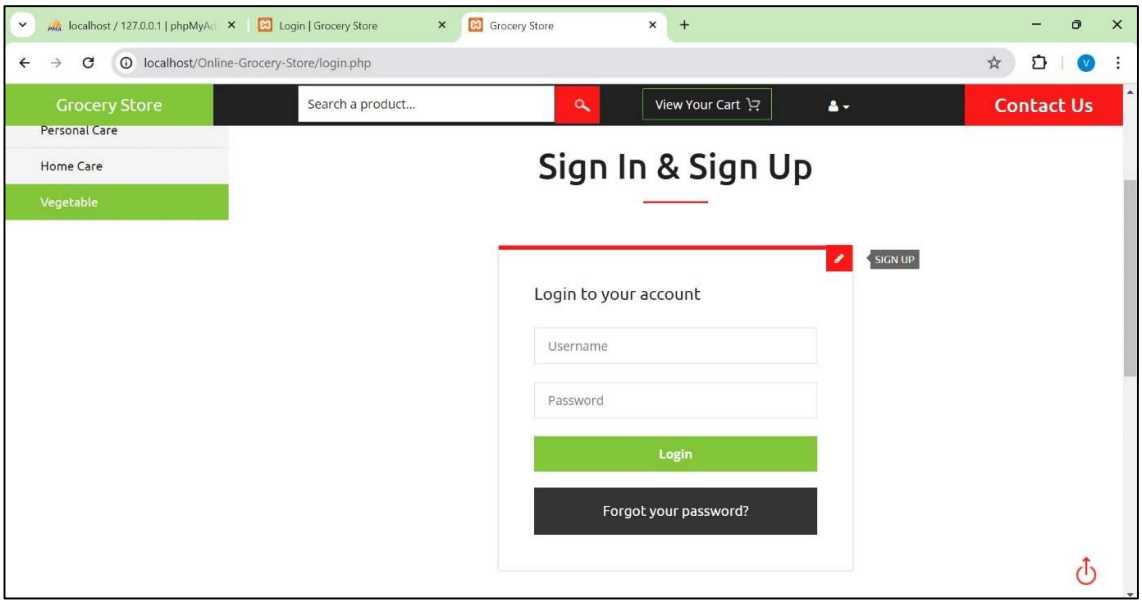

Fig 6.2: User Signin Page

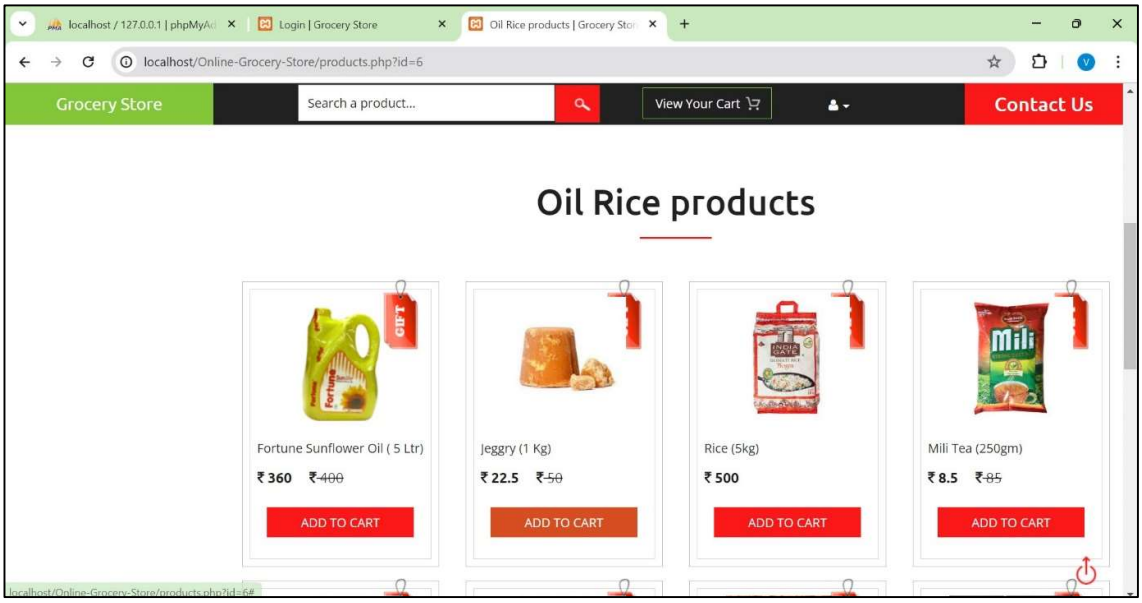

Fig 6.3: List of products

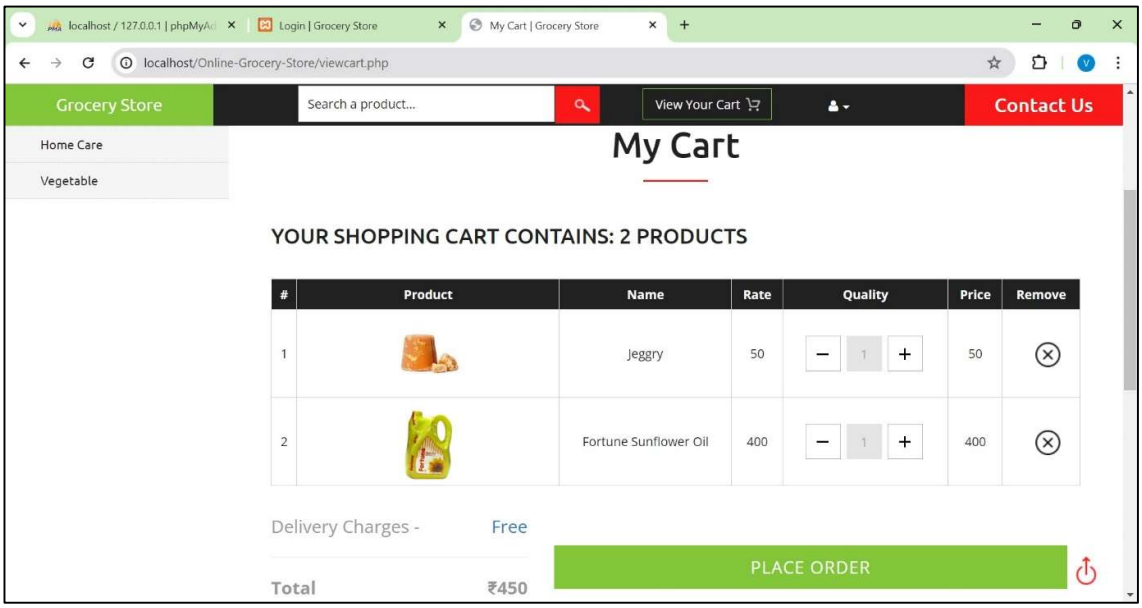

Fig 6.4: Cart

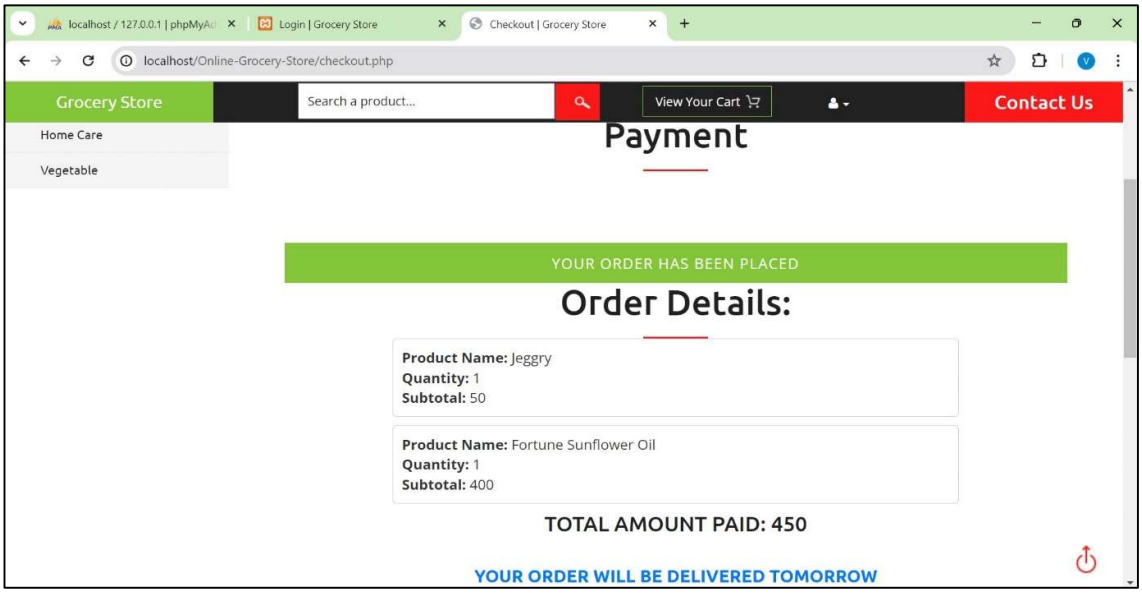

Fig 6.5: Payment

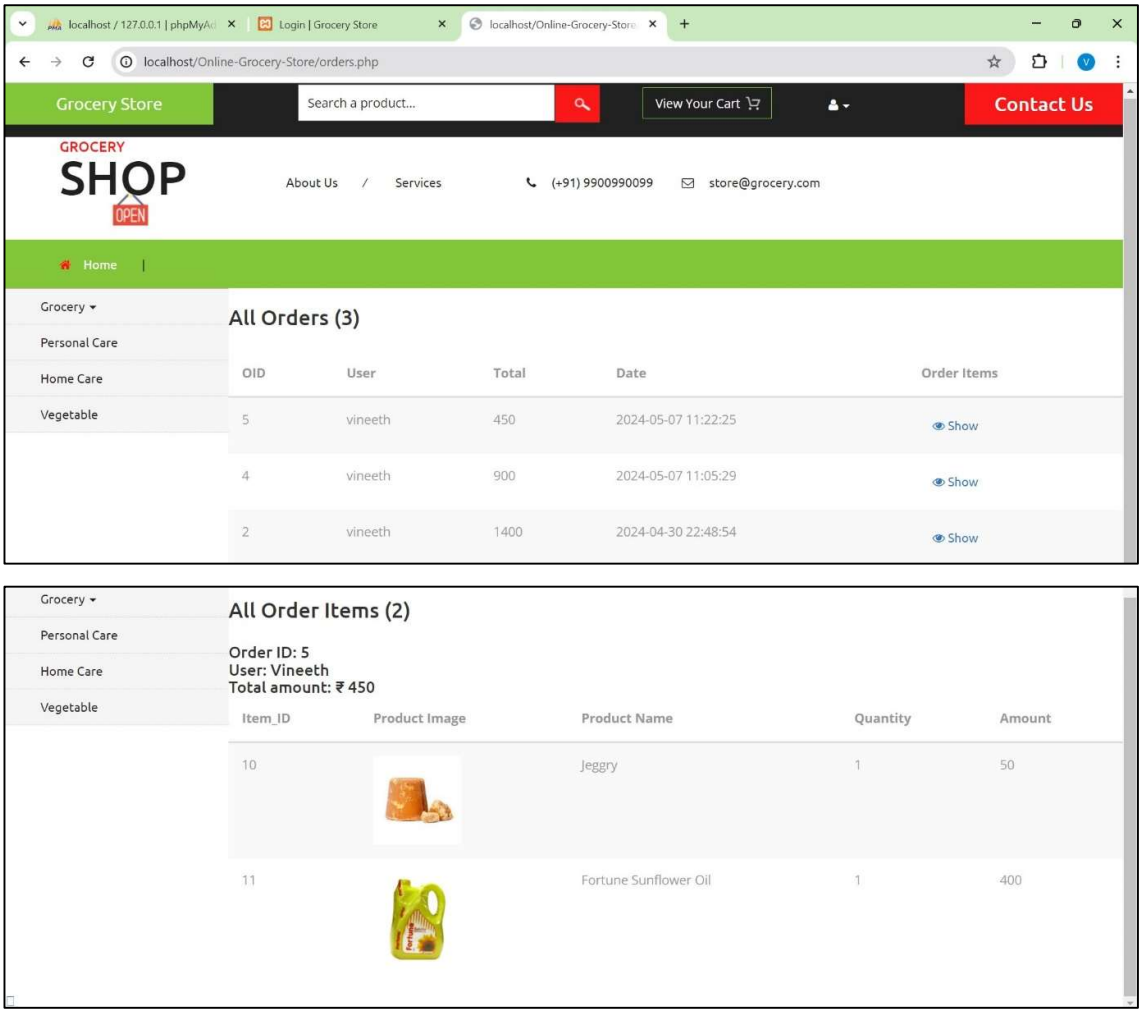

Fig 6.6: List of products in order history

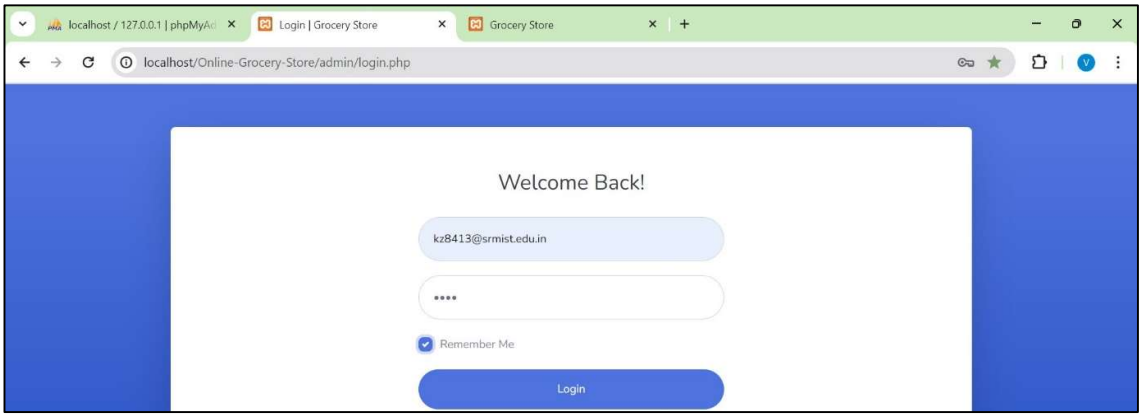

Fig 6.7: Admin Signin Page

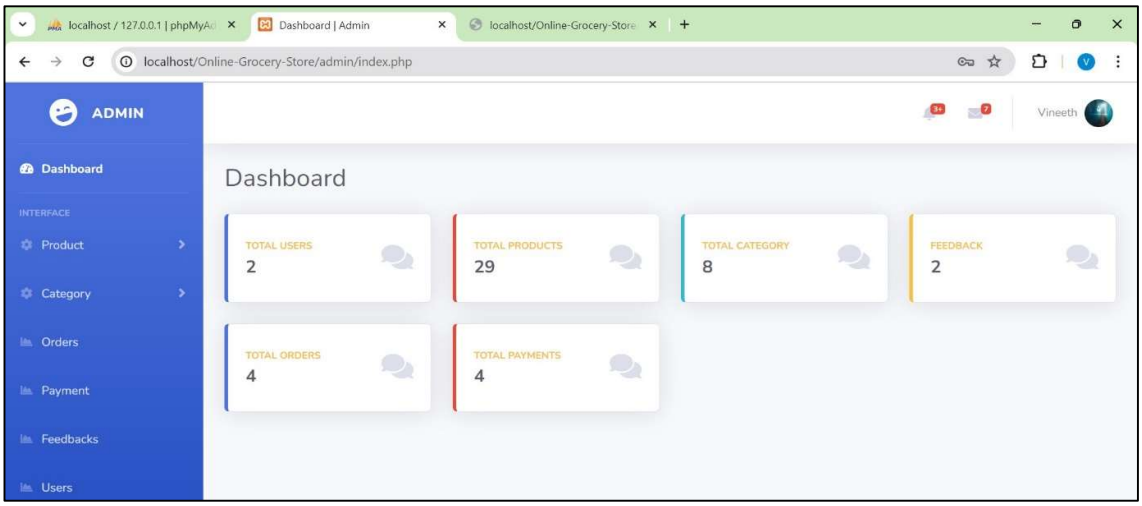

Fig 6.8: Admin Dashboard

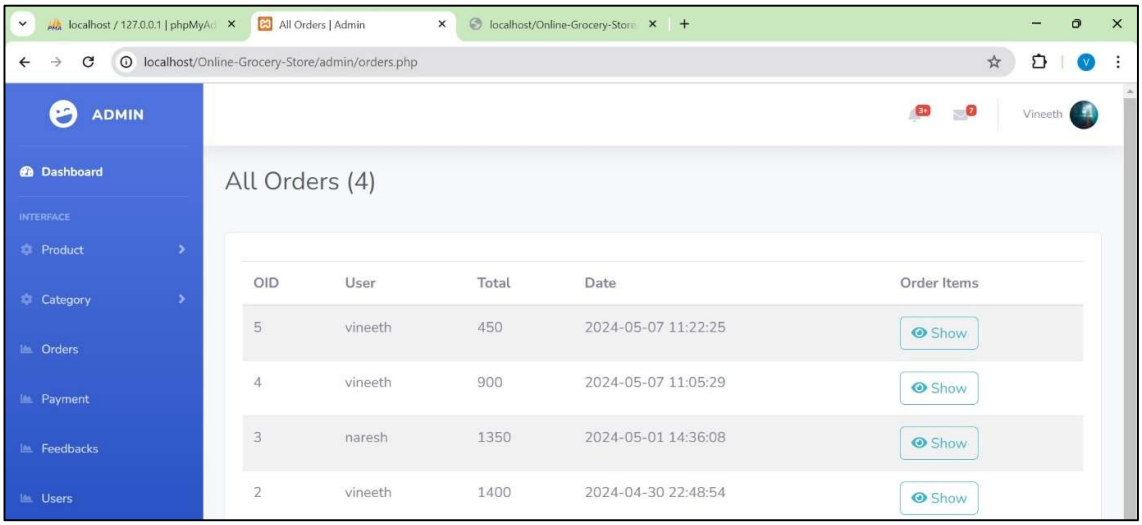

Fig 6.9: List of Orders in Admin Dashboard

CHAPTER 7
CONCLUSION AND FUTURE WORK

In this chapter, the conclusion that we have come to after completing this project has been written along with future work regarding this concept.

Conclusion

In conclusion, the development and implementation of the online grocery store web application have significantly transformed the landscape of grocery shopping, offering unparalleled convenience and accessibility to consumers. Through meticulous planning, robust design, and agile development methodologies, our team successfully created a user-friendly platform that caters to the diverse needs of modern shoppers. By leveraging advanced technologies such as AI-driven recommendation systems, secure payment gateways, and seamless user interfaces, we have not only streamlined the shopping experience but also fostered customer loyalty and satisfaction. The online grocery store web application stands poised for further evolution and expansion in response to changing market dynamics and consumer preferences. Continuous innovation, proactive customer engagement strategies, and strategic partnerships will be pivotal in sustaining growth and staying ahead of the competition.

Future Work

In considering future work for the online grocery store web application, several avenues present themselves for exploration and enhancement. Firstly, incorporating machine learning algorithms to personalize the shopping experience based on individual preferences and past purchasing behaviour could greatly enhance user satisfaction and loyalty. By analyzing data such as browsing history, the application can offer tailored product recommendations and promotions, thereby increasing customer engagement and sales. Additionally, optimizing the application for mobile devices and exploring opportunities for augmented reality (AR) or virtual reality (VR) shopping experiences could further enhance accessibility and engagement. Finally, ongoing efforts to improve backend infrastructure and security measures will be essential to support the continued growth and success of the online grocery store web application in the ever-evolving digital marketplace.

References

- https://www.w3schools.com/php/
- https://www.geeksforgeeks.org/sql-tutorial/
- https://www.w3schools.com/js/js_ajax_intro.asp
- https://jquery.com/
- https://getbootstrap.com/docs/5.3/examples/
- https://chatgpt.com/?oai-dm=1
- https://www.perplexity.ai/
- https://www.behance.net/search/projects/Grocery%20website
- https://www.w3schools.com/w3css/defaulT.asp
- https://www.w3schools.com/mysql/default.asp

Github Link: https://github.com/Vineethkolli/WDD_Online_Grocery_Store