### SIMATS SCHOOL OF ENGINEERING SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL

**SCIENCES CHENNAI-602105**

Fresh Farm Produce E-Commerce Platform: Streamlined Online Marketplace for Groceries

**A CAPSTONE PROJECT REPORT**

*Submitted in the partial fulfillment for the completion of the course*

**CSA4309 INTERNET PROGRAMMING FOR WEB SERVICES**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**Submitted by**

**YERABAPU DEEPTHI (192371040)**

**Under the Supervision of**

**L.Reetha**

**MAY 2025**

**DECLARATION**

I,**Yerabapu Deepthi** student of Bachelor of Engineering in the Department of Computer Science and Engineering, Saveetha Institute of Medical and Technical Sciences, Saveetha School of Engineering, Chennai, hereby declare that the work presented in this Capstone Project Work entitled **Fresh Farm Produce E-Commerce Platform: Streamlined Online Marketplace for Groceries** is the outcome of our own bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics.

**(Yerabapu Deepthi: 192371040)**

**CERTIFICATE**

This is to certify that the project entitled **“Automated Network Security Testing Tools”** submitted by **Nithin Kumar.V, Yuvaraj.R, Harish.S, Jaivignesh.S** has been carried out under my supervision. The project has been submitted as per the requirements in the current semester of B.E. Computer Science and Engineering.

Supervisor

L.Reetha

**Table of Contents**

| **S.NO** | **TOPICS** | **PAGE NO** |
| --- | --- | --- |
| **1.** | **Abstract** | **5** |
| **2.** | **Introduction** | **6** |
| **3.** | **Problem Statement:**  Consumers face limited access to fresh, locally-sourced farm produce due to inefficient distribution and lack of direct farmer-to-consumer platforms | **7** |
| **4.** | Architecture Diagram | 8 |
| **5.** | Description | 9 |
| **6.** | **Implementation** | **10-26** |
| **7.** | **Conclusion** | **27** |
| **8.** | **References** | **28** |

# ABSTRACT

The "Fresh Farm Produce E-Commerce Platform" is a digital marketplace connecting local farmers with customers to facilitate direct sales of fresh vegetables and fruits and also snacks . Built using HTML, CSS, JavaScript, and PHP, the platform provides a responsive and user-friendly interface for browsing, selecting, and ordering produce. Key features include a product catalog, advanced search, secure authentication, shopping cart, and order management. Farmers can list produce and manage sales, while customers enjoy real-time updates and secure payment options. The project promotes sustainability by reducing supply chain length, supporting local agriculture, and minimizing food waste. It fosters trust and transparency, enhancing convenience for consumers and empowering farmers in the digital era.

# INTRODUCTION

The "Fresh Farm Produce E-Commerce Platform" is an innovative online solution designed to bridge the gap between local farmers and consumers by facilitating the direct sale of fresh vegetables and fruits. In an era where convenience and accessibility are paramount, this platform empowers farmers to showcase their produce online while providing customers with a hassle-free shopping experience.

Developed using modern web technologies, the front end leverages HTML, CSS, and JavaScript to deliver a responsive and engaging interface, while the back end utilizes PHP to ensure efficient data handling and seamless functionality. The platform integrates essential features such as a comprehensive product catalog, search and filtering options, secure user authentication, and an intuitive shopping cart system to streamline the purchasing process.

This initiative aims to support local agriculture, promote sustainability, and reduce food waste by connecting producers directly with consumers. By shortening the supply chain, the platform not only ensures fresher produce but also fosters transparency, trust, and mutual growth for farmers and customers alike in the digital marketplace.

# PROBLEM STATEMENT

Access to fresh, high-quality vegetables and fruits is a common challenge for consumers due to inefficiencies in traditional supply chains. These supply chains involve multiple intermediaries who inflate costs, reduce profits for farmers, and cause delays that affect produce freshness. As a result, customers often pay higher prices for lower-quality products.

For farmers, the lack of direct access to consumers limits their ability to earn fair prices and expand their market reach. Many small-scale farmers rely heavily on middlemen, which restricts their profitability and hampers their growth. In addition, the absence of modern tools to manage sales, inventory, and customer interactions creates further obstacles.

Consumers, too, face limited options for buying directly from local farmers. Most existing marketplaces lack transparency and fail to provide a streamlined experience for selecting, purchasing, and receiving fresh produce. The disconnect between farmers and consumers prevents the adoption of sustainable practices that could benefit both parties.

Environmental concerns also arise from traditional distribution systems, including increased food waste and carbon emissions due to long supply chains and storage times.

To address these challenges, there is a need for a robust online platform that connects farmers and customers directly. This platform should empower farmers to list their produce, set prices, and manage orders efficiently, while giving customers the ability to browse and purchase fresh produce easily. By reducing intermediaries, such a platform can ensure fair pricing, promote local agriculture, minimize waste, and encourage sustainability.

A modern e-commerce platform leveraging technologies like HTML, CSS, JavaScript, and PHP can provide a practical solution, streamlining the process for both farmers and consumers while fostering trust and transparency in the fresh produce market.

# ARCHITECTURE DIAGRAM

The architecture of the "Fresh Farm Produce E-Commerce Platform" is designed to ensure seamless interaction between users (customers and farmers), the application logic, and data storage. The architecture is divided into three main layers:

## Presentation Layer (Front-End):

The Presentation Layer provides the user interface for interacting with the platform. It is accessible via devices like desktops, tablets, and smartphones. This layer uses HTML, CSS, and JavaScript to create a responsive, visually appealing, and user-friendly interface. It allows users to browse products, search for items, register or log in, and place orders. Dynamic elements like search filters, shopping cart, and form validations are handled here to enhance user experience.

## Application Layer (Back-End):

The Application Layer serves as the backbone of the platform, processing all requests and implementing business logic. This layer is built using PHP and handles operations such as user authentication, order processing, and management of product listings. It validates input from the front-end and interacts with the database to store or retrieve data as required. Additionally, this layer ensures secure payment integration and maintains session management for user activities.

## Database Layer:

The Database Layer is implemented using MySQL and serves as the repository for all platform data. It stores user details (customer and farmer profiles), product information (name, price, category, and stock availability), and order records (order ID, status, and payment details). The database supports efficient data retrieval and advanced query execution, such as searching for products or retrieving order histories.

## Data Flow:

The workflow starts with users interacting with the Presentation Layer to perform actions such as searching, ordering, or updating account details. These requests are sent to the Application Layer, where the PHP server processes them and communicates with the Database Layer for data retrieval or updates. The Application Layer then sends the response back to the Presentation Layer, ensuring the user receives real-time updates and feedback.

This layered architecture ensures scalability, reliability, and a streamlined user experience, enabling the platform to effectively connect farmers with consumers while promoting sustainability and efficiency.

# HTML & CSS CODE

**LOGIN PAGE**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Login</title>

<link rel="stylesheet" href="styles.css">

<style>

/\* General Styles \*/

body {

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

background: url('backg.jpg') no-repeat center center fixed;

/\* Replace with your image path \*/

background-size: contain;

/\* Cover the entire viewport \*/

font-family: 'Times New Roman', serif;

background-color: #fffbf6; /\* Warm beige background \*/

}

.container {

text-align: center;

border: 2px solid #ffa07a; /\* Orange border \*/

padding: 30px;

border-radius: 10px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

background-color: rgba(255, 255, 255, 0.9); /\* Slightly transparent white for readability \*/

max-width: 400px;

width: 100%;

}

</style>

</head>

<body>

<div class="container">

<h2>Login</h2>

<form onsubmit="login(event)">

<input type="email" placeholder="Email ID" id="loginEmail" required><br>

<div class="error" id="loginEmailError"></div>

<input type="password" placeholder="Password" id="loginPassword" required><br>

<div class="error" id="loginPasswordError"></div>

<a href="forgotpassword.html" class="forgot-password">Forgot Password?</a>

</form>

<p>Don't have an account? <a href="register.html">Register</a></p>

</div>

<script>

function login(event) {

event.preventDefault(); // Prevent form submission

const email = document.getElementById('loginEmail').value.toLowerCase();

const password = document.getElementById('loginPassword').value;

// Check credentials (mock implementation)

const storedEmail = localStorage.getItem('email');

const storedPassword = localStorage.getItem('password');

if (email === storedEmail && password === storedPassword) {

alert('Login successful!');

window.location.href = 'home.html'; // Redirect to home page

} else {

document.getElementById('loginEmailError').textContent = 'Invalid email or password.';

document.getElementById('loginPasswordError').textContent = 'Invalid email or password.';

}

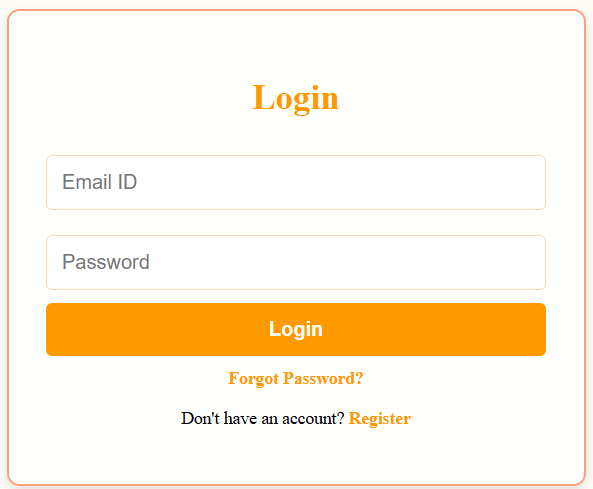
}

</script>

</body>

</html>

# OUTPUT

****

**REGISTER PAGE**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Register</title>

<link rel="stylesheet" href="styles.css">

<style>

body {

background-image: url('backg.jpg');

background-size: cover;

background-position: center;

background-repeat: no-repeat;

background-attachment: fixed;

margin: 0;

padding: 0;

font-family: 'Arial', sans-serif;

color: #666;

display: flex;

align-items: center;

justify-content: center;

height: 100vh;

}

.container {

width: 430px;

padding: 30px;

background: rgba(255, 255, 255, 0.533);

border-radius: 10px;

box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);

text-align: center;

border: 2px solid #ff9900;

}

return valid;

}

function validateEmail(email) {

const lowerCaseEmail = email.toLowerCase();

const regex = /^[a-z0-9.\_%+-]+@[a-z0-9.-]+\.[a-z]{2,4}$/;

if (email !== lowerCaseEmail) {

return false;

}

return regex.test(lowerCaseEmail);

}

function validatePassword(password) {

const regex = /^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)(?=.\*[@$!%?&]).{8,}$/;

return regex.test(password);

}

alert('Successfully registered!');

window.location.href = 'login.html';

}

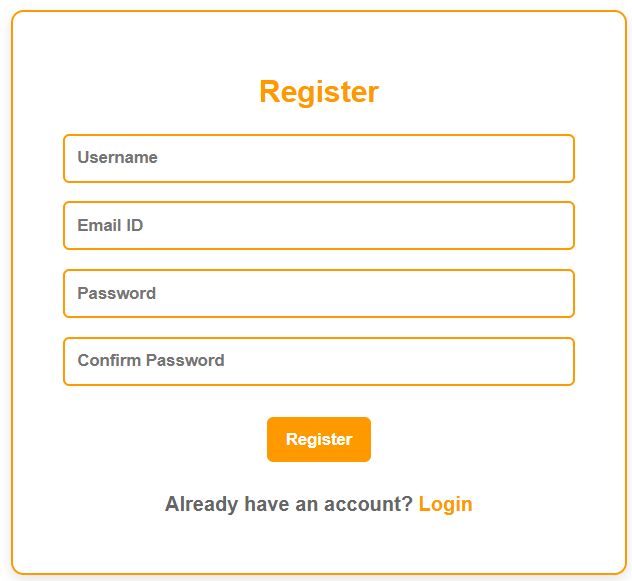
});

</script>

</body>

</html>

# OUTPUT

****

**FRUITS PAGE**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Fruits - Grocery Store</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f4f4f4;

margin: 0;

padding: 0;

}

header {

background-color: #333;

color: white;

padding: 15px 0;

text-align: center;

}

nav a {

color: white;

margin: 0 15px;

text-decoration: none;

}

</div>

<div class="product">

<img src="strawberry.jpg" alt="Strawberry">

<h3>Strawberry</h3>

<p>$3.00 per pound</p>

<button class="btn" onclick="addToCart('Strawberry', 3.00)">Add to Cart</button>

</div>

<div class="product">

<img src="kiwi.jpg" alt="Kiwi">

<h3>Kiwi</h3>

<p>$1.50 each</p>

<button class="btn" onclick="addToCart('Kiwi', 1.50)">Add to Cart</button>

</div>

<div class="product">

<img src="pineapple.jpg" alt="Pineapple">

<h3>Pineapple</h3>

<p>$3.50 each</p>

<button class="btn" onclick="addToCart('Pineapple', 3.50)">Add to Cart</button>

</div>

<div class="product">

<img src="mango.jpg" alt="Mango">

<h3>Mango</h3>

<p>$1.80 each</p>

<button class="btn" onclick="addToCart('Mango', 1.80)">Add to Cart</button>

</div>

<div class="product">

<img src="blueberry.jpg" alt="Blueberry">

<h3>Blueberry</h3>

<p>$4.00 per pint</p>

<button class="btn" onclick="addToCart('Blueberry', 4.00)">Add to Cart</button>

</div>

</div>

<div class="cart-container">

<h2>Your Cart</h2>

<ul id="cartItems"></ul>

<p id="totalCost">Total: $0.00</p>

<button onclick="proceedToPayment()">Proceed to Payment</button>

</div>

<footer>

<p>&copy; 2024 Grocery Store. All rights reserved.</p>

</footer>

totalCost.textContent = `Total: $${total.toFixed(2)}`;

}

function proceedToPayment() {

window.location.href = 'payment.html';

}

displayCart(); // Show cart on page load

</script>

</body>

</html>

# OUTPUT

### 

### VEGETABLES PAGE

# <!DOCTYPE html>

# <html lang="en">

# <head>

# <meta charset="UTF-8">

# <meta name="viewport" content="width=device-width, initial-scale=1.0">

# <title>Vegetables - Grocery Store</title>

# <style>

# body {

# font-family: Arial, sans-serif;

# background-color: #f4f4f4;

# margin: 0;

# padding: 0;

# }

# header {

# background-color: #333;

# color: white;

# padding: 15px 0;

# text-align: center;

# }

# nav a {

# color: white;

# margin: 0 15px;

# text-decoration: none;

# }

# nav a:hover {

# text-decoration: underline;

# }

# <div class="product">

# <img src="spinach.jpg" alt="Spinach">

# <h3>Spinach</h3>

# <p>$1.00 per bunch</p>

# <p>Rich in iron and vitamins, great for salads or cooking!</p>

# <button class="btn" onclick="addToCart('Spinach', 1.00)">Add to Cart</button>

# </div>

# <div class="product">

# <img src="brinjal.jpg" alt="Brinjal">

# <h3>Brinjal</h3>

# <p>$0.90 each</p>

# <p>Vitamin-rich and versatile in dishes!</p>

# <button class="btn" onclick="addToCart('Brinjal', 0.90)">Add to Cart</button>

# </div>

# </div>

# <div class="cart">

# <h2>Your Cart</h2>

# <ul id="cartItems"></ul>

# <p id="totalCost">Total: $0.00</p>

# <button class="proceed-btn" onclick="proceedToPayment()">Proceed to Payment</button>

# </div>

# <footer>

# <p>&copy; 2024 Grocery Store. All rights reserved.</p>

# </footer>

# cartItems.appendChild(li);

# });

# totalCost.textContent = `Total: $${total.toFixed(2)}`;

# }

# function proceedToPayment() {

# window.location.href = 'payment.html';

# }

# displayCart(); // Show cart on page load

# </script>

# </body>

# </html>

# OUTPUT

### 

### SNACKS PAGE

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Grocery Store Snacks</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f4f4f4;

margin: 0;

padding: 20px;

}

header {

text-align: center;

background-color: #333;

color: white;

padding: 15px 0;

}

.snack-item img {

max-width: 100%;

height: auto;

border-radius: 5px;

}

.cost {

font-weight: bold;

color: green;

}

.add-to-cart {

background-color: #007bff;

color: white;

border: none;

padding: 10px;

border-radius: 5px;

cursor: pointer;

margin-top: 10px;

}

.add-to-cart:hover {

background-color: #0056b3;

}

position: relative;

bottom: 0;

width: 100%;

}

</style>

<script>

let cart = JSON.parse(sessionStorage.getItem('cartItems')) || [];

let total = parseFloat(sessionStorage.getItem('cartTotal')) || 0;

function addToCart(itemName, itemCost) {

cart.push({ name: itemName, cost: itemCost });

total += itemCost;

updateCartInSession();

displayCart();

alert(`${itemName} has been added to your cart.`);

}

</script>

</head>

<body>

<header>

<h1>Grocery Store Snacks</h1>

<nav>

<a href="home.html">Home</a>

<a href="fruits.html">Fruits</a>

<a href="vegetables.html">Vegetables</a>

<a href="cold-beverage.html">Cold Beverages</a>

<a href="payment.html">payment</a>

</nav>

</header>

<div class="snack-container">

<div class="snack-item">

<img src="chips.jpg" alt="Chips">

<h2>Chips</h2>

<p class="cost">$1.50</p>

<p>Crunchy and delicious!</p>

<button class="add-to-cart" onclick="addToCart('Chips', 1.50)">Add to Cart</button>

</div>

<div class="snack-item">

<img src="cookies.jpg" alt="Cookies">

<h2>Cookies</h2>

<p class="cost">$2.00</p>

<p>Sweet and satisfying!</p>

<button class="add-to-cart" onclick="addToCart('Cookies', 2.00)">Add to Cart</button>

</div>

<div class="snack-item">

<img src="popcorn.jpg" alt="Popcorn">

<h2>Popcorn</h2>

<div class="cart">

<h2>Your Cart</h2>

<ul id="cartItems"></ul>

<p id="totalCost">Total: $0.00</p>

</div>

<footer>

<p>&copy; 2024 Grocery Store. All rights reserved.</p>

</footer>

<script>

// Display the cart on page load

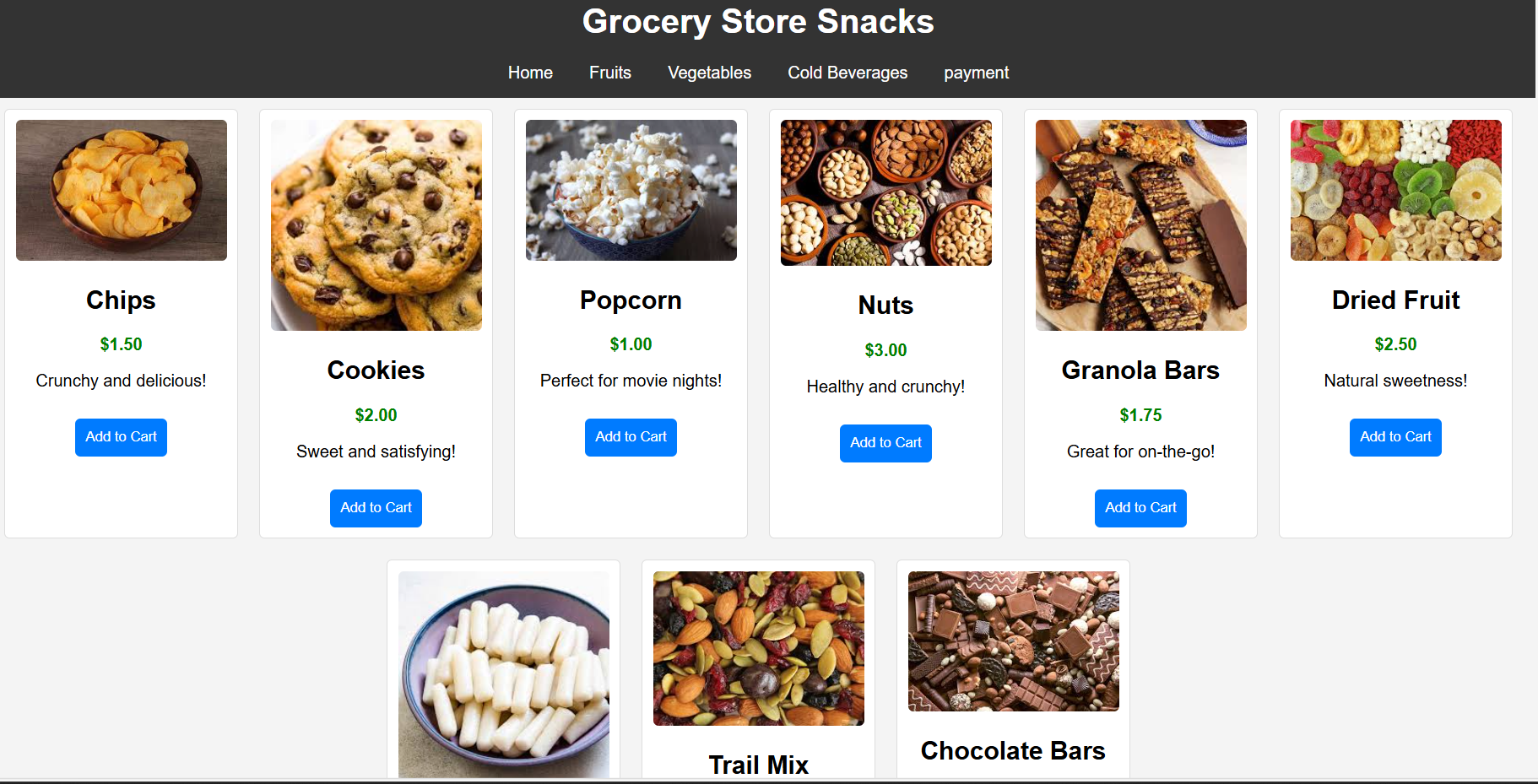
displayCart();

</script>

</body>

</html>

# OUTPUT

****

**PAYMENT PAGE**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Payment - Grocery Store</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f4f4f4;

margin: 0;

padding: 20px;

}

h1 {

text-align: center;

}

.payment-container {

background: white;

padding: 20px;

border-radius: 5px;

max-width: 600px;

margin: 0 auto;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

button:hover {

background-color: #0056b3;

}

.total-cost {

font-weight: bold;

margin: 15px 0;

text-align: center;

}

.cart-summary {

margin-top: 20px;

border-top: 1px solid #ddd;

padding-top: 10px;

}

.cart-item {

display: flex;

justify-content: space-between;

padding: 5px 0;

}

.back-button {

// Clear cart

sessionStorage.removeItem('cartTotal');

sessionStorage.removeItem('cartItems');

window.location.href = 'home.html'; // Redirect to home or confirmation page

});

// Function to go back to the previous page

function goBack() {

window.history.back();

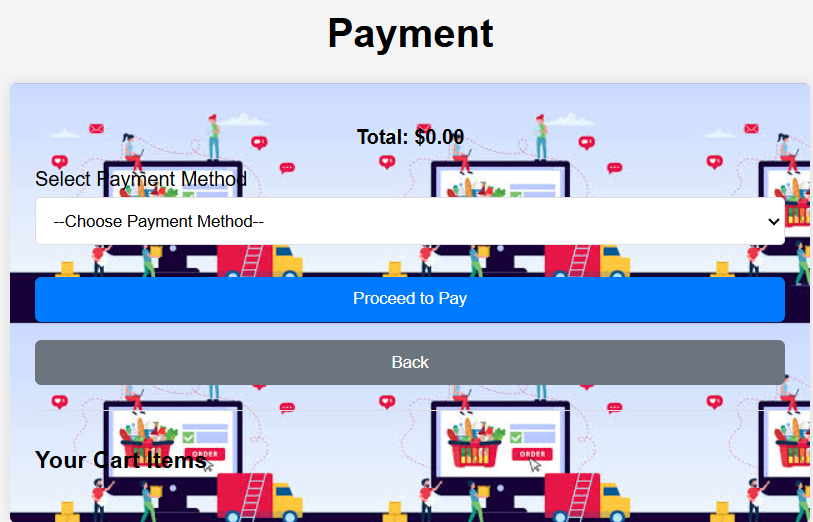
}

</script>

</body>

</html>

**OUTPUT**

****

**CONCLUSION**

The "Fresh Farm Produce E-Commerce Platform" is an innovative solution designed to bridge the gap between local farmers and consumers, providing a seamless, direct marketplace for fresh vegetables and fruits. By leveraging modern web technologies such as HTML, CSS, JavaScript for the front-end and PHP for the back-end, the platform offers a user-friendly experience, secure transactions, and efficient management of orders and products. The integration of a MySQL database ensures effective data storage and retrieval, supporting a dynamic, real-time interaction between users and the system.

This platform not only enhances convenience for consumers by offering fresh produce directly from farms, but also empowers farmers by providing them with a broader customer base and better control over their sales. Furthermore, it promotes sustainable practices by reducing the supply chain, minimizing food waste, and supporting local agriculture. In conclusion, this e- commerce platform is a significant step toward revolutionizing the way fresh produce is purchased and sold, fostering trust, transparency, and efficiency in the marketplace.

# REFERENCES

#### Mann, S., & Wadhwa, A. (2020). E-commerce website design for agriculture and farm products. Journal of Agricultural Informatics, 11(2), 45-56.

This paper discusses the design and functionality of e-commerce websites specifically for agriculture and farm products.

#### Oke, S. O., & Akinlolu, F. O. (2020). Sustainable supply chain management in agriculture: E-commerce solutions for farmers. International Journal of Sustainable Agriculture, 8(3), 202-210.

Explores sustainable practices in agriculture and how e-commerce solutions can help in reducing supply chain inefficiencies.

#### Jain, R., & Agarwal, P. (2021). Modern web technologies in e-commerce applications: A case study of PHP-based platforms. International Journal of Web Technologies, 5(1), 39-47.

Provides insights into how modern web technologies, including PHP, are used in e-commerce applications, with a focus on agricultural platforms.

#### Kumar, P., & Saha, S. (2021). Database management systems for e-commerce websites: Best practices for scalability and security. Journal of Data Management and Security, 12(4), 314-328.

This reference discusses the role of database management systems in ensuring the scalability and security of e-commerce websites.

#### Sinha, S., & Tiwari, R. (2019). E-commerce platforms for agriculture: Bridging the gap between farmers and consumers. Agricultural Engineering Review, 6(1), 75-82.

Focuses on how e-commerce platforms help bridge the gap between farmers and consumers, improving both access to products and market efficiency.