Software Development for Web Apps Sessional

Course Code: CSE-410

Project Report

Group Members

Maisha Binte Alam, ID: 1111018

Mojahidul Islam, ID: 1111023

Koli Rani Pal, ID: 1111028

Nusrat Jahan Lima, ID: 1111029

Course Teacher's Name: Rabeya Sultana

Department of Computer Science and Engineering (CSE)

Bangladesh Army International University of Science & Technology (BAIUST)

Sharatnagar, Syedpur 3501, Bangladesh

Team Members

Photo	Name	ID	Email	Contact NO.	Total Credit Passed	CGPA acquired
	Maisha Binte Alam	1111018	maisha18baiust@gmail.com	01556773127	135.25	3.35
	Mojahidul Islam	1111023	mojahidulislam-sagor1234@gmail.com	0135565969	135.25	
	Koli Rani Pal	1111028	palkoli581@gmail.com	01310032460	135.25	
	Nusrat Jahan Lima	1111029	nusrat.ja-han168241@gmail.com	01748576454	135.25	3.64

Executive Summary

The **"Green Grocers"** Grocery Shop project is a cutting-edge e-commerce initiative aimed at transforming the conventional grocery shopping paradigm. Developed through the integration of HTML, CSS, JavaScript, and PHP, the platform provides customers with a seamless online experience for procuring fresh products without the need to physically visit the store. The website boasts a user-friendly interface, a comprehensive product catalog, and secure transaction processing. Through user registration and login features, customers can personalize their accounts, facilitating order tracking and future transactions.

This project addresses the contemporary demand for convenient and time-saving solutions, positioning itself as a forward-thinking response to the evolving needs of modern consumers. The successful implementation of **Green Grocers** lays the foundation for potential expansions, technological advancements, and ongoing enhancements, promising a dynamic and responsive platform for the future.

Contents

List of Figures

List of Tables

Introduction

1.1 Goals and Objectives of the project

1.1.1 Goals

The primary goal of the **"Green Grocers"** Grocery Shop is to streamline the shopping experience for customers by offering a digital platform to purchase fresh items.

- Design and implement an intuitive user interface that allows customers to navigate seamlessly through the website, facilitating a user-friendly and efficient shopping experience.
- Ensure that customers have access to a comprehensive product catalog of fresh items, categorized and described in detail to aid in informed purchasing decisions.
- Implement robust security measures to safeguard customer information during transactions, assuring a secure and trustworthy platform for online shopping.
- Provide features such as user registration, order tracking, and personalized user accounts to enhance customer convenience and build a long-lasting relationship.

1.1.2 Objectives

Our objectives include:

- Create an extensive catalog of fresh products, including detailed descriptions, images, and prices, ensuring customers can make well-informed decisions.
- Develop a user registration and login system to personalize the shopping experience, allowing customers to track orders, save preferences, and expedite future transactions.
- Create a responsive web design that adapts to various devices, including desktops, tablets, and smartphones, optimizing the shopping experience across different platforms.
- Incorporate a feedback and review system, allowing customers to share their experiences and provide valuable insights for continuous improvement.

1.2 Scope of the work

The scope of this project encompasses the development of an e-commerce website with the following key features:

- User registration and login functionality.
- Product catalog with detailed descriptions.
- Shopping cart and checkout process.
- Integration with XAMPP for database management.

1.3 System overview

The system comprises a front-end interface developed using HTML, CSS, and JavaScript, ensuring an engaging and interactive user experience. The backend is powered by PHP for server-side processing, with XAMPP serving as the database management system.

1.4 Terms, Acronyms, and Abbreviations Used

- HTML: Hypertext Markup Language
- CSS: Cascading Style Sheets
- JavaScript: Scripting language for web development
- PHP: Hypertext Preprocessor
- XAMPP: Cross-platform Apache, MySQL, PHP

Project Management Plan

2.1 Project Organization

The "Green grocery" website is envisioned as a one-stop destination for individuals seeking a wide variety of fresh and organically grown fruits and vegetables,grocery items. This e-commerce platform will offer a seamless shopping experience, complete with a user-friendly interface and a secure payment system. Users will be able to browse, select, and purchase their desired produce from the comfort of their homes, ensuring they receive the best quality products.

2.1.1 Individual Contribution to the project

Individual Contribution to the project are shown in Table-2.1

Member Name	Requirement Specification	Planning	Designing	Model Building	User Interface	Testing	Deployment
Koli rani pal	✓						
Nusrat jahan	✓						
Maisha Alam Prova	✓						
Mojahidul Islam Sagor	✓						

Table 2.1: Individual Contribution to the project

2.2 Process Model Used

Choosing the right process model for grocery shop website project depends on various factors such as project size, complexity, and the level of flexibility we needed. the features we are adding to our project we consider using an Agile development process. Agile is well-suited for projects that require flexibility, continuous feedback, and the ability to adapt to changing requirements. Here is the reason we choose to use Agile process-

1.Flexibility to Changing Requirements:

Websites often undergo changes in requirements, especially as stakeholders gain more clarity or as market demands evolve. Agile's adaptive nature allows for seamless integration of these changes during the development process.

2.Incremental Development:

Agile promotes the delivery of a minimum viable product (MVP) early in the project, followed by incremental releases of additional features. This aligns well with the typical phased approach to website development, allowing stakeholders to see tangible progress quickly.

3.Frequent Customer Feedback:

Agile emphasizes regular collaboration with stakeholders, and in a website project, this translates to continuous feedback on design, functionality, and user experience. Frequent iterations enable the team to adjust and refine features based on real-time input.

4.Iterative Design and Development:

Websites often involve complex user interfaces and design elements. Agile allows for iterative design and development, ensuring that the visual and functional aspects of the website are refined incrementally over the course of the project.

5.Cross-Functional Teams:

Agile encourages the formation of cross-functional teams, where members have diverse skills. In a website project, this might include developers, designers, UX/UI specialists, and content creators working collaboratively to address various aspects of the website.

6.Regular Reviews and Retrospectives:

Agile ceremonies like sprint reviews and retrospectives are valuable in website development. Regular reviews ensure that the product aligns with expectations, and retrospectives allow the team to reflect on what went well and what can be improved.

7.Continuous Integration and Testing:

Agile promotes continuous integration practices, which is essential in web development to ensure that code changes are regularly integrated and tested. Automated testing further enhances the reliability of the website.

8.Time-Boxed Sprints:

Agile projects are typically organized into time-boxed iterations called sprints. This helps in planning and managing work effectively. For a website project, sprints can be used to focus on specific features or components.

9.Prioritization of Features:

Agile allows for the prioritization of features based on their importance and value. This ensures that the most critical aspects of the website are developed and delivered first.

10.Collaborative Decision-Making:

Agile promotes collaborative decision-making within the team. In web development, decisions regarding design, functionality, and technical choices are often collaborative efforts, and Agile supports this cooperative approach.

11.Customer Satisfaction:

With frequent releases, continuous communication, and the ability to adapt to changing requirements, Agile contributes to higher customer satisfaction. Clients see tangible progress and have the flexibility to guide the development based on their evolving needs.

12.Adaptation to Emerging Technologies:

Websites often leverage emerging technologies. Agile's adaptability allows teams to incorporate new tools or technologies as they become available or as project requirements demand.

2.2.1 Rationale for choosing lifecycle model

Selecting the appropriate life cycle model for grocery shop website project is a critical decision that impacts how the project will be planned, executed, and controlled. The choice should be based on various factors including project requirements, the level of flexibility needed, and the development team's expertise. Here are the rationales for choosing the Agile methodology for this project:

1.Flexibility in Requirements:The grocery shop website project involves features like registration, login, product browsing, reviews, and payment options. These requirements may evolve or change during the development process due to market trends or stakeholder feedback. Agile is known for its flexibility and adaptability to changing requirements. It allows for continuous feedback and adjustments, making it well-suited for projects where requirements are not completely known or may change over time.

2.Incremental Development: The website's features, such as product browsing, reviews, and payment options, can be developed incrementally.Agile promotes incremental and iterative development, delivering a minimum viable product (MVP) early and then adding features in successive iterations.

3.Customer Involvement: In a grocery shop website, user experience and customer satisfaction are paramount. Regular collaboration with stakeholders, including end-users, is essential to ensure the website meets their expectations.Agile methodologies prioritize customer involvement throughout the development process. Regular feedback and reviews ensure that the delivered product aligns closely with the users' needs and preferences.

4.Rapid Time-to-Market: The grocery shop industry is competitive, and a quicker time-to-market provides a strategic advantage.Agile's iterative and incremental approach allows for faster delivery of functional components. This enables the team to respond quickly to market demands and deliver value to customers sooner.

5.Cross-Functional Collaboration: Building a grocery shop website involves various

disciplines, including frontend and backend development, UI/UX design, and possibly mobile responsiveness.Agile encourages the formation of cross-functional teams, where members with diverse skills collaborate effectively. This helps in addressing different aspects of the website simultaneously.

6.Continuous Integration and Testing: Ensuring the website's reliability and performance is crucial for an online platform dealing with transactions and user data. Agile methodologies emphasize continuous integration and testing. This ensures that code changes are regularly integrated, and automated testing is performed, maintaining the overall quality of the website.

7.Risk Management:The grocery shop website project may face uncertainties and risks, such as security concerns or integration challenges.Agile incorporates risk management as part of its iterative approach. Regular assessments and adaptations allow the team to identify and mitigate risks early in the project.

2.3 Risk Analysis

Identifying and managing risks is a crucial aspect of project management. Here are some potential risks associated with our grocery shop website project:

1.Security Concerns:

Possibility of data breaches, unauthorized access, or other security vulnerabilities.Implement robust security measures, use encryption for sensitive data, and conduct regular security audits we can solve this issue.

2.Integration Challenges:

Difficulties in integrating third-party services or payment gateways.Thoroughly research and test integration points early in the development process. Work closely with service providers can solve this problem.

3.Changing Requirements:

Stakeholders may change their requirements during the development process.Regular communication with stakeholders, well-defined requirements, and a flexible development approach (such as Agile) to accommodate changes is a suitable solution for this type of problems.

4.Scalability Issues:

The website may face performance issues as the user base grows. By designing the system architecture to be scalable, conduct performance testing, and have a plan for scaling infrastructure if needed can solve this issue.

5.User Adoption:

Users may not find the website intuitive or may resist using new features.Conducting user testing, gather feedback, and make iterative improvements. Provide user training if necessary are the step we can take.

6.Data Loss:

Loss of user or transaction data due to system failures or other issues.By implementing regular backups, disaster recovery plans, and data validation checks we can avoid these type of problems.

7.Technical Debt:

Rushed development may lead to suboptimal code quality and technical debt.Prioritize code quality, conduct code reviews, and allocate time for refactoring as needed are the solutions.

8.Budget Overruns:

Unexpected expenses leading to exceeding the project budget. Regularly monitor project costs, identify potential risks early, and have contingency plans in place are the solution we may take.

9.Mobile Responsiveness:

Inadequate mobile responsiveness may lead to a poor user experience on mobile devices.We should Prioritize mobile responsiveness in the design phase and conduct thorough testing across various devices.

10.Market Competition:

Changes in the market or competitive landscape may impact the success of the grocery shop.We should Regularly monitor the market, stay informed about industry trends, and be prepared to adapt the business strategy.

2.4 Constraints to project implementation

(Schedule, Budget, Software & Hardware constraints)

Implementing a grocery shop website project may face various constraints that can impact the schedule, budget, and technical aspects of the project. Identifying and understanding these constraints is crucial for effective project management. Here are some potential constraints to consider:

Schedule Constraints:

1.Time-to-Market Pressure: There may be a need to launch the website within a specific timeframe to meet market demands or seasonal considerations.This constraint can affect the development schedule, potentially leading to trade-offs between features and time.

Budget Constraints:

1.Limited Funding:The project may have budgetary restrictions, limiting the resources available for development, marketing, and ongoing maintenance. Budget constraints may influence decisions on technology choices, feature prioritization, and overall project scope.

2.Unforeseen Costs:Unexpected expenses, such as additional development requirements or changes in market conditions, may arise during the project.Unanticipated

costs can strain the project budget and require adjustments to resource allocation. Software Constraints:

Technology Stack Compatibility: The choice of technologies may be constrained by existing systems, legacy infrastructure, or compatibility requirements.Compatibility issues may arise if the chosen technologies do not integrate seamlessly with existing systems.

Hardware Constraints:

Scalability Challenges:The website's architecture and infrastructure may face challenges in scaling to accommodate increasing user traffic.Scalability constraints can affect performance and user experience, requiring careful consideration of server resources and load balancing.

Device Compatibility: The need to support a wide range of devices and screen sizes may pose challenges in ensuring a consistent user experience.Design and development efforts must account for device compatibility, potentially impacting the project timeline.

Market Competition:The competitive landscape may influence the project's features and marketing strategy.Pressure to stay competitive may affect the project's schedule and feature prioritization.

Human Resource Constraints:

Team Expertise:The availability and expertise of the project team members may impact the choice of technologies and the speed of development.

2.5 Hardware and Software Resource (Tools/Language) Requirements

Frontend Technologies:

1.HTML,CSS: Essential for structuring and styling web pages.

JavaScript: Core scripting language for interactive web pages.

Bootstrap: CSS framework for responsive and mobile-first front-end development.

Backend Technologies:php.

Database Management: MySQL

Version Control:

Git:Distributed version control system for tracking changes in source code during development.

2.6 Project Timeline and Schedule

Creating a project timeline and schedule involves breaking down the project into specific tasks, estimating the time required for each task, and organizing these tasks

into a coherent timeline. Below is a generalized project timeline for our grocery shop website project.The actual timeline may vary based on project-specific requirements, team expertise, and other factors.

Initiation Phase (Week 1):

Define project scope and objectives. Conduct initial market research and competitor analysis.

Planning Phase (Week 2):

Develop a detailed project plan outlining tasks, responsibilities, and timelines.Create a budget and allocate resources.Define the technology stack and infrastructure requirements.

Design and Development Phase (Week3-4):

Develop wireframes and mockups for key pages (homepage, product listing, user profile).Finalize the user interface (UI) and user experience (UX) design. Set up the development environment and version control. Implement core features such as user registration, login, and homepage.Develop the product listing, categorization, and search functionality.

Testing Phase (Week):

Conduct unit testing for individual components. Perform integration testing to ensure seamless communication between different modules. Implement user acceptance testing (UAT) to gather feedback from users.Address and resolve identified issues and bugs.

Deployment Phase (Week):

Prepare for the production environment, Conduct final testing in the production environment.Monitor for any issues and implement necessary fixes.

Continuously monitor website performance and user feedback. Implement updates, optimizations, and additional features based on user needs. Conduct regular security audits and apply necessary patches. Provide ongoing customer support and address any issues promptly.

2.7 Estimated Budget

Estimated Budget:Green Grocery Shop Website Project

Project Initiation and Planning:

Market Research:5,000 tk

Competitor Analysis:3,000 tk

Project Planning: 7,000 tk

Design Phase:

Wireframing and Mockups:10,000 tk

UI/UX Design: 15,000 tk

Development Phase:

Frontend Development: 30,000 tk

Backend Development: 35,000 tk

Database Setup: 8,000 tk

Third-Party Integrations: 10,000 tk

Testing Phase:

Unit Testing: 5,000 tk

Integration Testing: 7,000 tk

User Acceptance Testing (UAT): 10,000 tk

Deployment Phase:

Server Setup and Configuration: 5,000 tk

Production Deployment: 7,000 tk

Launch and Marketing:

Marketing Strategy: 15,000 tk

SEO Implementation: 8,000 tk

Launch Campaigns: 10,000tk

Post-Launch Optimization and Maintenance (First Year):

Ongoing Development: 40,000 tk

Security Audits and Updates: 15,000tk

Customer Support: 20,000 tk

Total Estimated Budget: 248,000 tk

2.8 Social/Cultural/Environmental impact of the project

(Include a description of what impact your project will have on individuals and society) The development and implementation of a grocery shop website can have several social, cultural, and environmental impacts, affecting both individuals and society at large.

Social Impact:

1.Convenience and Accessibility:

The project's primary goal is to provide a convenient platform for individuals to purchase groceries online, enhancing accessibility for people with busy schedules,

mobility challenges, or those living in remote areas.It contributes to improved access to essential goods, particularly benefiting individuals who may have difficulty accessing physical stores.

2.Community Engagement:

By incorporating features such as reviews and user interactions, the website can foster a sense of community among users. People can share experiences, recommend products, and engage in discussions about their preferences.It creates a virtual community around grocery shopping, enhancing the social aspect of an otherwise mundane activity.

3.Employment Opportunities:

The project may create job opportunities, both directly (website development, customer support) and indirectly (in the logistics and delivery sector).Job creation contributes positively to local economies and provides individuals with employment opportunities.

Cultural Impact:

1.Diverse Product Offerings:

The website can offer a wide range of products, including culturally specific or region-specific items, catering to the diverse needs and preferences of users.It supports cultural diversity by providing access to a variety of products that align with different cultural backgrounds.

2.Localization of Services:

The project can incorporate localization features, region-specific promotions, to make the platform more culturally relevant to users from different areas.It promotes inclusivity and ensures that the website is culturally sensitive to the diverse backgrounds of its users.

3.Preservation of Local Businesses:

The project can collaborate with local vendors, supporting small businesses and promoting locally sourced products. This contributes to the preservation of local cultural practices and helps sustain traditional businesses.

Environmental Impact:

1.Reduced Carbon Footprint:

By providing an online platform for grocery shopping, the project has the potential to reduce the need for individual trips to physical stores, thus lowering carbon emissions associated with transportation.It supports environmental sustainability by encouraging a more eco-friendly approach to obtaining groceries.

2.Efficient Supply Chain Management:

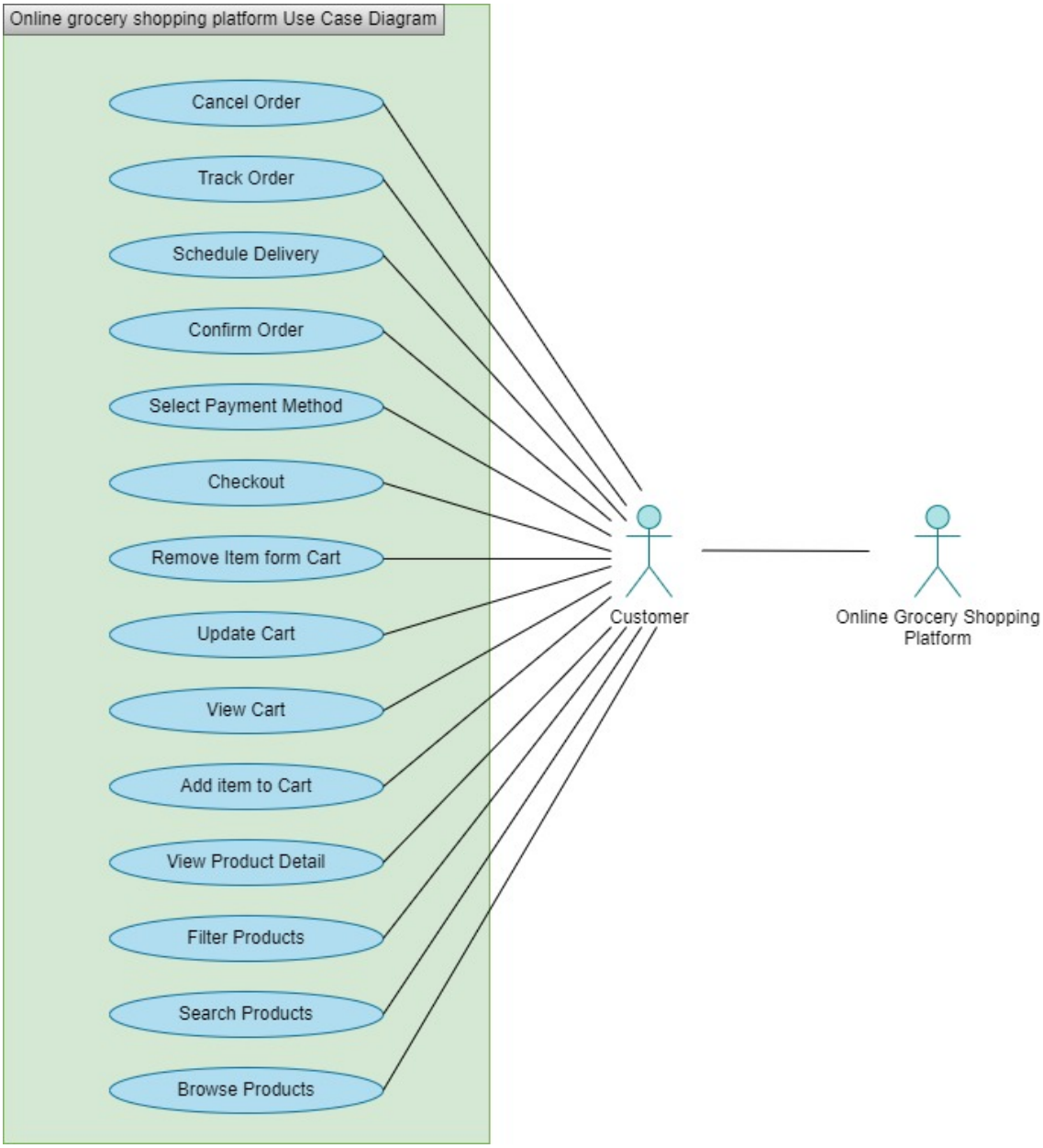
The project can implement efficient supply chain practices, minimizing wastage and optimizing inventory management.This reduces environmental impact by minimizing food waste and promoting more sustainable resource use.

3.Encouraging Eco-Friendly Practices:

The project can promote eco-friendly products or packaging options, encouraging users to make environmentally conscious choices.It raises awareness about sustainable practices and fosters a sense of responsibility towards the environment among users.

Requirement Specifications

3.1 Use case diagram with Graphical and Textual Description

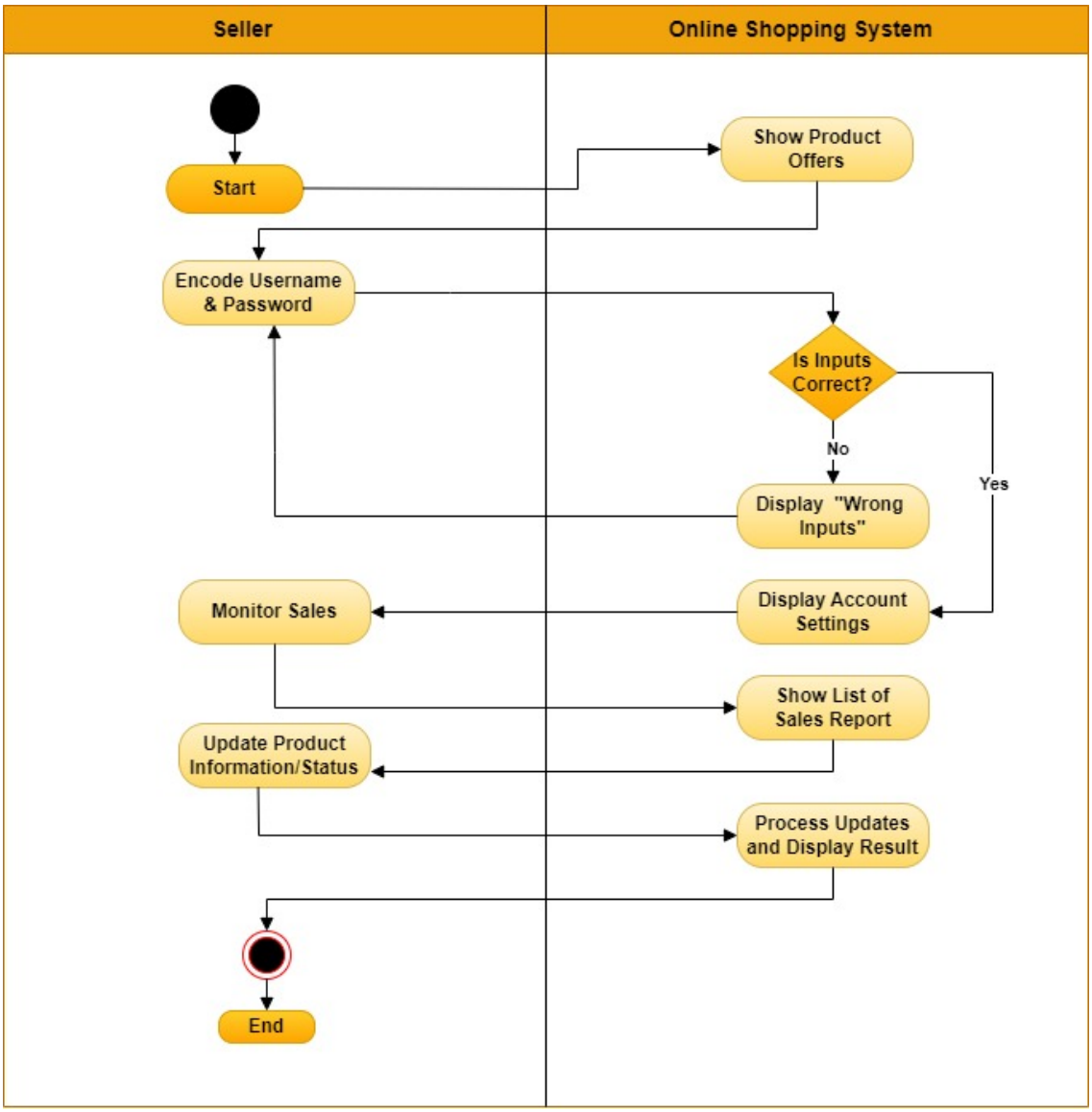


Textual Description:
An online grocery shopping platform is a software solution that allows customers to purchase groceries online. The use case diagram for the platform outlines fourteen use cases, such as canceling orders, tracking orders, confirming orders, checking out, updating the cart, adding items to the cart, filtering products, and searching products. These use cases are essential for managing the customer's shopping experience and ensuring that they receive their groceries accurately and efficiently.

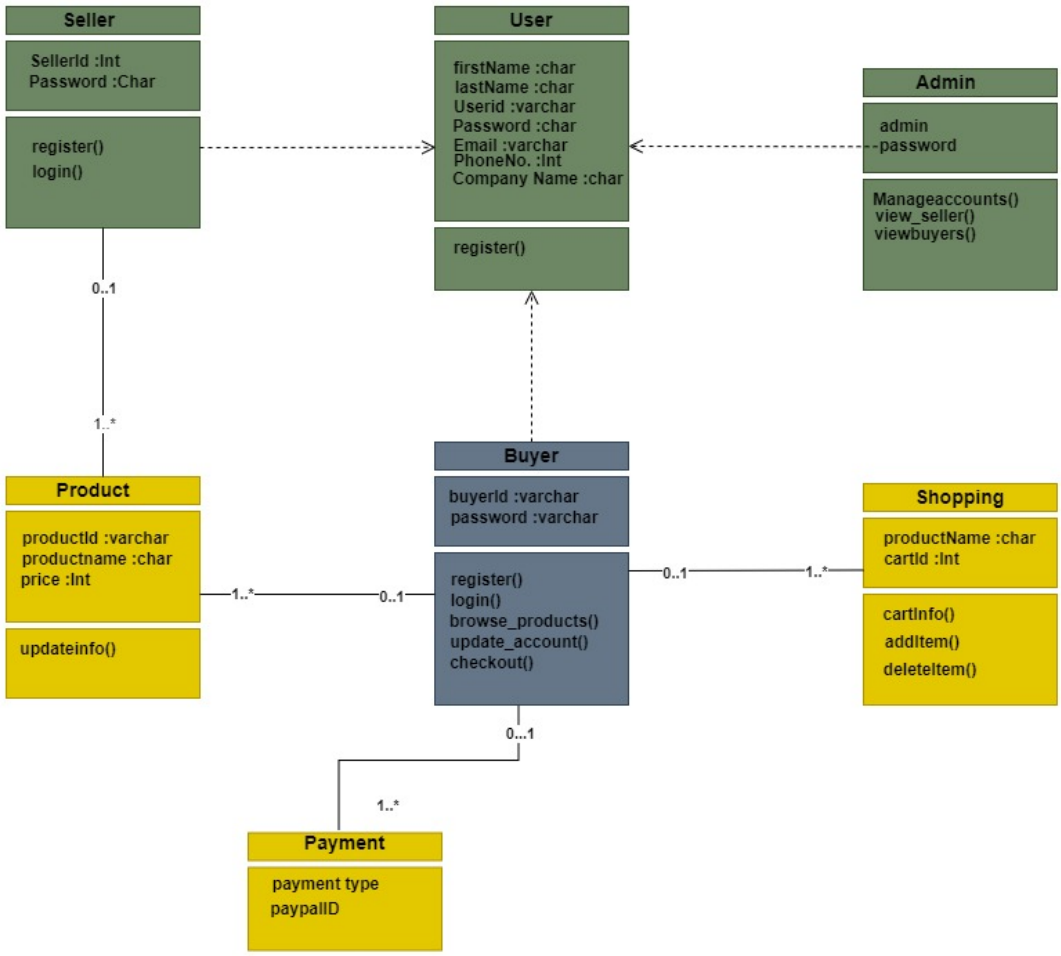
Canceling orders is important for providing customers with flexibility and ensuring that they have a positive shopping experience. Tracking orders is essential for ensuring that customers receive their groceries on time and can plan accordingly. Confirming orders involves reviewing the order details and ensuring that the correct items and quantities are included, while checking out is the final step in the purchasing process and is essential for ensuring that customers receive their groceries accurately and efficiently.

Managing the customer's shopping experience is critical to the success of an online grocery shopping platform. Updating the cart, adding items to the cart, filtering products, and searching products allow customers to manage the contents of their shopping cart and search for and view the available products. These use cases are essential for providing customers with a smooth shopping experience and increasing sales.

3.2 Activity Diagram

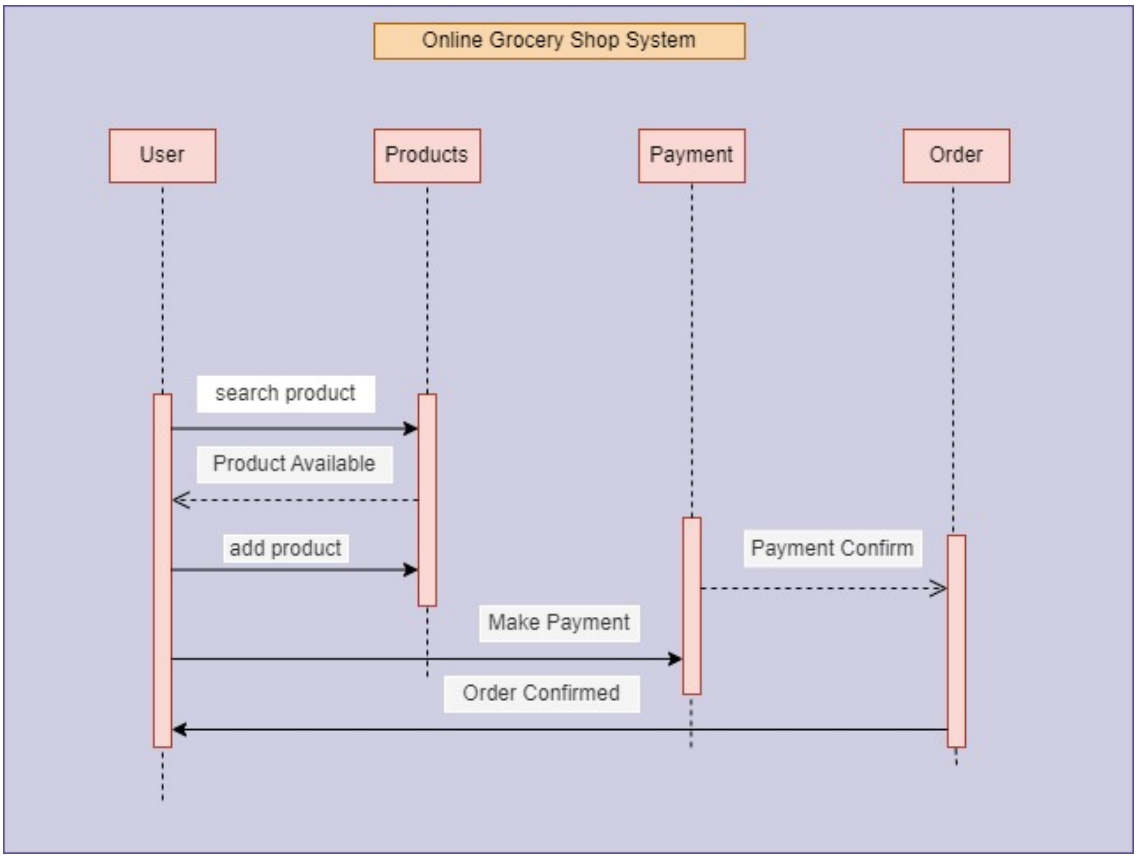


3.3 Static model – class diagram



Static model – class diagram

3.4 Dynamic model – sequence diagram



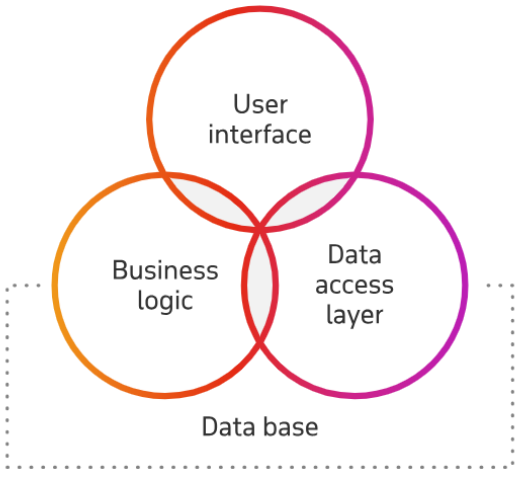
Architecture

4.1 Architectural model/style used

The architectural model or style for an online grocery shop website design can be chosen based on various factors, including scalability, performance, maintainability, and specific business requirements. Here are some commonly used architectural models/styles for designing online grocery shop websites:

Monolithic Architecture:
Designed in line with the monolithic architecture model, this ecommerce solution is a unified and one-piece mechanism, regardless of the number of in-built features. A monolithic ecommerce solution connects to an enterprise database and typically encompasses a user interface (for a mobile or a web app), a data access layer, and a business logic layer. Monolithic architecture is a traditional way of developing software that is considered slightly outdated today but still has some time-proven advantages as well as drawbacks.

An example of a monolithic ecommerce architecture

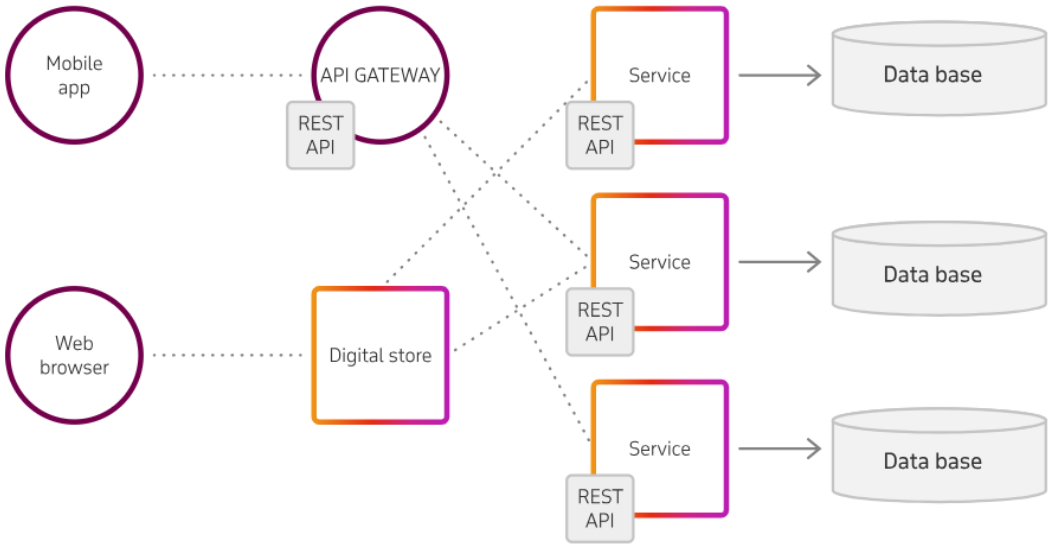


On the one hand, the development of a monolithic solution typically doesn't take much time, in part because of the straightforward and quick testing process. But on the other hand, the too-close connection between all system components and their integrity makes any functional updates quite risky, since by changing one component, you can unintentionally affect the entire system and jeopardize its integrity

and stability.

Microservices Architecture:
A microservices-based ecommerce architecture consists of a set of interconnected services called microservices that communicate with each other via multiple API interfaces. Each microservice operates as a subsystem with its own business logic and architecture, and this is the main advantage of microservices.
Opting for microservices, you can quickly and painlessly modify and replace any element of your solution and mitigate risks related to the integrity and performance of your entire software system. On the other hand, the abundance of independent components requires continuous coordination within the development team, which makes the building of a microservices architecture quite a complex process.

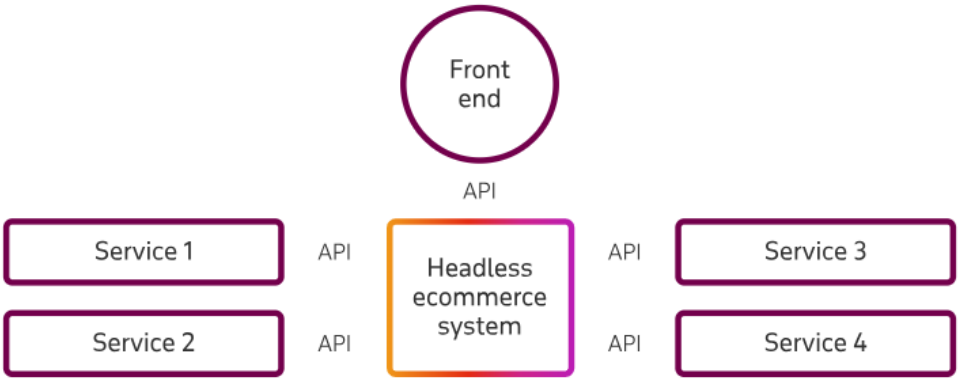
An example of a microservices ecommerce architecture



Microservices architecture involves breaking down the application into smaller, independent services that communicate with each other through APIs. Each microservice is responsible for a specific business capability, promoting scalability and ease of maintenance.

Headless Architecture:
A headless architecture is one of the most popular versions of the microservices concept. Here, the digital system is divided into two separate layers, back-end, and front-end, connected via an API. This architecture is typically used by enterprises that want to tap into additional digital commerce channels and be able to distribute content across them quickly.

An example of a headless ecommerce architecture

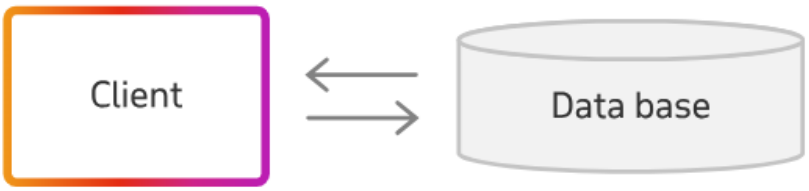


In a headless architecture, the frontend (head) and backend are decoupled, allowing for flexibility in delivering content to various devices and channels. This approach can be beneficial for providing a seamless user experience across different platforms.

Two-tier architecture:
Depending on how many subsystems your solution consists of (for example, client, logic, or database layers), it can also have a two-tier, three-tier, or multi-tier architecture. In other words, all solutions, including microservices-based digital systems and monolithic apps, can simultaneously be two-, three-, and multi-tier.

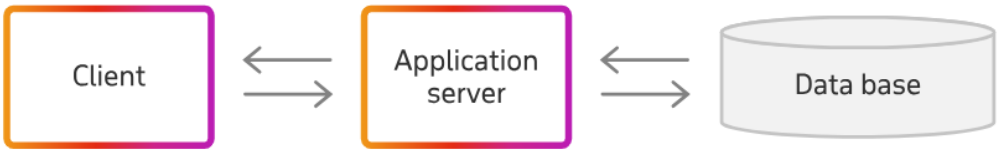
Typically, a two-tier architecture consists of a client-server and a database. Since you don't need to design and test many software elements, two-tiered apps are pretty easy to develop. But at the same time, since they can't handle many user requests and spikes in traffic, two-tiered solutions aren't easily scaled.

A two-tier architecture



Three-tier architecture:
Having, in turn, a three-tier architecture, ecommerce systems encompass three levels: the UI, the application layer, and the database, which stores and processes system data. Being logically separated, each level can be developed in parallel. This allows companies to scale their solutions quickly and allocate the right talent to the needs of a particular project. But on the other hand, given the large number of components a three-tier architecture entails, retailers have to scale up their development processes as well, potentially complicating project and personnel management.

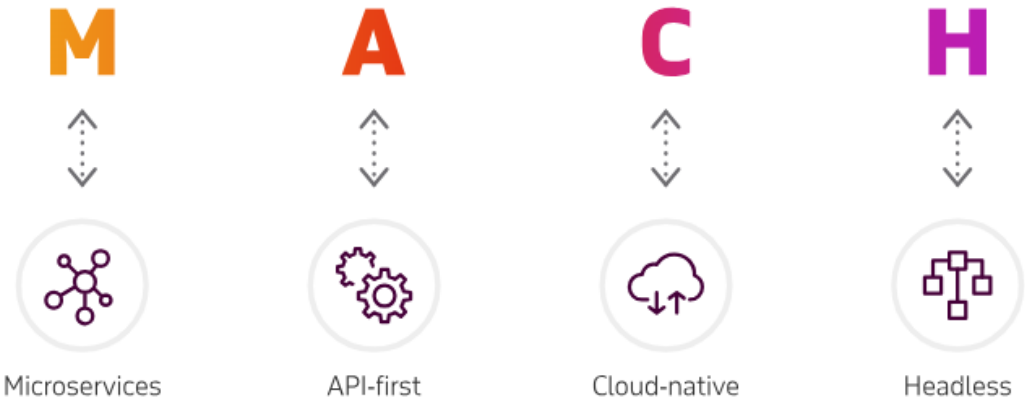
A three-tier architecture



Given that both types of software architecture have their distinct pros and cons, choosing between a two-tier and three-tier architecture will likely depend on the requirements and scope of your project. The main thing to remember here is that the more levels your system includes, the more complex and challenging the development may become.

The MACH architecture:
This innovative concept emerged only in 2018 and is gaining popularity among enterprises striving to establish a future-proof and flexible ecommerce architecture. The term MACH stands for Microservices, API-first, Cloud-native, and Headless, and this architecture enables the adopters to make quick software upgrades, thereby providing enterprises with the maximum possible technological flexibility and adaptability. However, such technological advancements come at a price, because once you adopt the MACH architecture, you will significantly complicate most of your IT-management processes.

The MACH architecture



So, if you have the resources and expertise to carry out a development project that involves several technologies (such as cloud and microservices), implementing MACH can prove truly transformative for your business agility and customer engagement. In the first place, given that the architecture can consist of many microservices, each written in its own language, companies can quickly add the necessary functionality, for example, establish multi-channel commerce or expand their range of services. At the same time, due to the cloud-based deployment model, you can easily scale the computing capabilities of your ecommerce solution and support the growth and development of your business.

Service-Oriented Architecture (SOA):
SOA is an architectural pattern where different services communicate with each other over a network. It emphasizes the use of loosely coupled, interoperable services that can be combined to meet business needs.

Progressive Web App (PWA):
PWAs leverage modern web technologies to deliver a native app-like experience. They are designed to be fast, responsive, and reliable, providing offline capabilities and push notifications.

Serverless Architecture: Serverless architecture involves outsourcing server management to a third-party provider. Functions are executed in response to events, and developers are billed based on actual usage. This approach can reduce infrastructure management overhead.

Event-Driven Architecture:
An event-driven architecture involves the use of events to trigger and communicate between services. This can be beneficial for real-time updates, order processing, and other dynamic interactions in an online grocery shop.

E-commerce Platform:
Some online grocery shops may opt to use existing e-commerce platforms or frameworks tailored to handle the specific requirements of selling products online. Examples include Magento, WooCommerce, Shopify, and others.

Containerization and Orchestration:
Containerization (using tools like Docker) and orchestration (using tools like Kubernetes) can be employed to package and deploy applications consistently across different environments, providing scalability and ease of management. The choice of architecture depends on factors such as the scale of the project, development team expertise, scalability requirements, and the need for flexibility and maintainability. It's not uncommon for hybrid approaches or combinations of these architectural styles to be used based on specific use cases within the online grocery shop ecosystem.

4.2 Technology, software, and hardware used

The technology, software, and hardware used for designing an online grocery shop website can vary based on factors such as project requirements, scale, budget, and the preferences of the development team. Here is a general overview of the components involved:

Technology Stack

Frontend Development:
HTML/CSS/JavaScript: Standard web technologies for creating the user interface.

Angular, or Vue.js: Popular frontend frameworks for building dynamic and responsive user interfaces.

Bootstrap or Materialize: Frontend frameworks for responsive design.

Backend Development:
Programming Language: Choices may include Node.js (JavaScript), PHP.

Server Framework: Express for Node.js, Flask for Python, Spring Boot for Java, etc.

APIs: RESTful APIs or GraphQL for communication between frontend and backend.

Database

Relational Database: MySQL

Server-Side Rendering (SSR) or Static Site Generation (SSG): Depending on the chosen frontend framework can be used for better performance.

Authentication and Authorization: OAuth or JWT for user authentication and authorization. Identity providers like Auth0 or Firebase Authentication.Payment Processing:Integration with payment gateways such as Stripe, PayPal, or Braintree for secure transactions.

Search Engine Optimization (SEO): SEO-friendly practices and tools for optimizing the website's visibility on search engines.

Software

E-commerce Platform: Custom-built solution or using existing e-commerce platforms like Magento, WooCommerce (WordPress), Shopify, or others.

Content Management System (CMS): Vs, drawio or other CMS for managing non-product content.

Version Control: Git for version control, GitHub or GitLab for code repository hosting.

Development Environment: Integrated Development Environment (IDE) like Visual Studio Code, or others.

Hardware

Web Servers: Apache, Microsoft Internet Information Services (IIS) for serving web content.

Cloud Services: Hosting on cloud platforms like Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform for scalability and reliability.

Database Servers: Dedicated servers or managed database services on the chosen cloud platform.

Design

5.1 Component level design following pattern

Component design is a critical phase in software development, where the architectural blueprint of a system transforms into tangible, modular entities. Each component represents a self-contained, reusable unit with a specific set of functionalities, fostering a structured and scalable approach to building software. In the context of an online grocery shop, component design involves breaking down the system into distinct elements such as user interfaces, back-end services. This modular approach not only enhances maintainability and code re-usability but also facilitates collaborative development and easier troubleshooting. Here is the brief description of Component level design of the Green Grocery shop.

1. User Interface Components:

1.1 Homepage Component: Responsible for displaying featured products, promotions, and categories.Interacts with the product catalog Register,cart,contact,login,search bar.Allows users to navigate to different sections of the platform.

1.2 Product Catalog Component: Displays a list of available products with details.Connects with the backend to fetch product information.

1.3 User Account/login Component: Handles user authentication and authorization. Manages user profiles, order history, and preferences. Allows users to update personal information and password.

1.4 Contact Give users the contact info of the authorities for any emergency need.

1.5 Shopping Cart Component: Manages the items selected by the user for purchase. Allows users to update quantities, remove items, and proceed to checkout.Integrates with the backend for real-time updates.

1.6 Totl price Show total bill of the shopping

1.7 Search bar Allow users to search their desire item from the product list.

All these above part are header component.

1.8 Checkout Component: Manages the order confirmation and payment process. Integrates with external payment gateways for secure transactions. it's a part of cart section.

1.9 Delivery brand Show all type of brand that are available

1.10 Categories show all the product category.

2. Backend Components:

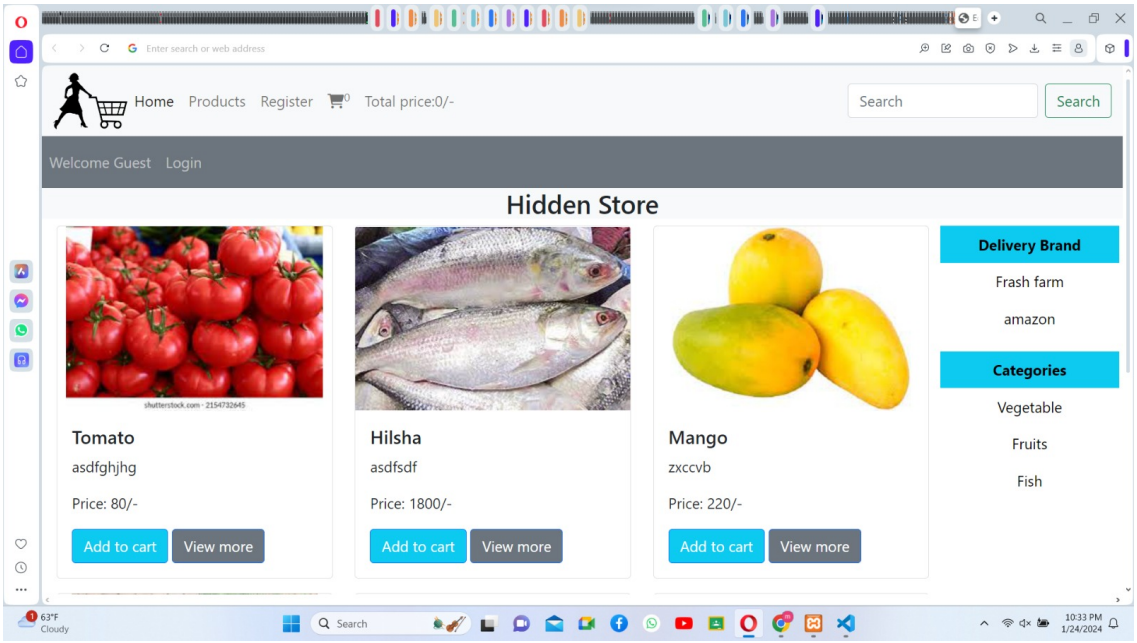
2.1 Product Component: Manages the product database.

2.2 Category: It has all the product category data.

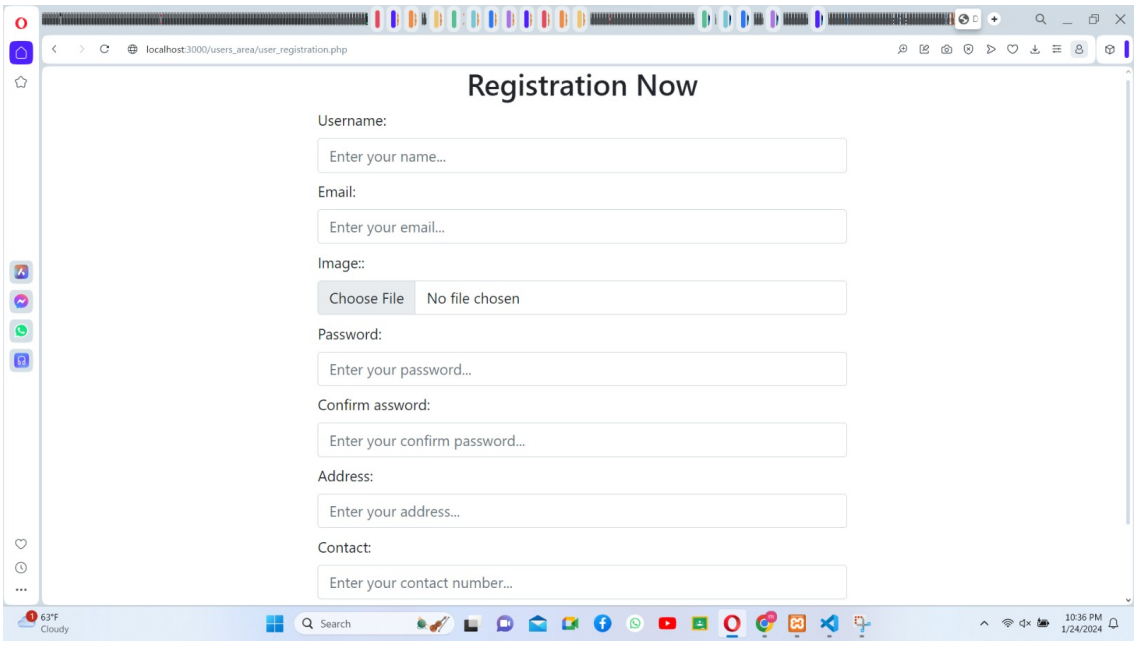
2.3 User Authentication Component: Handles user and authorities registration, login, and logout.

5.2 GUI (Graphical User Interface) design

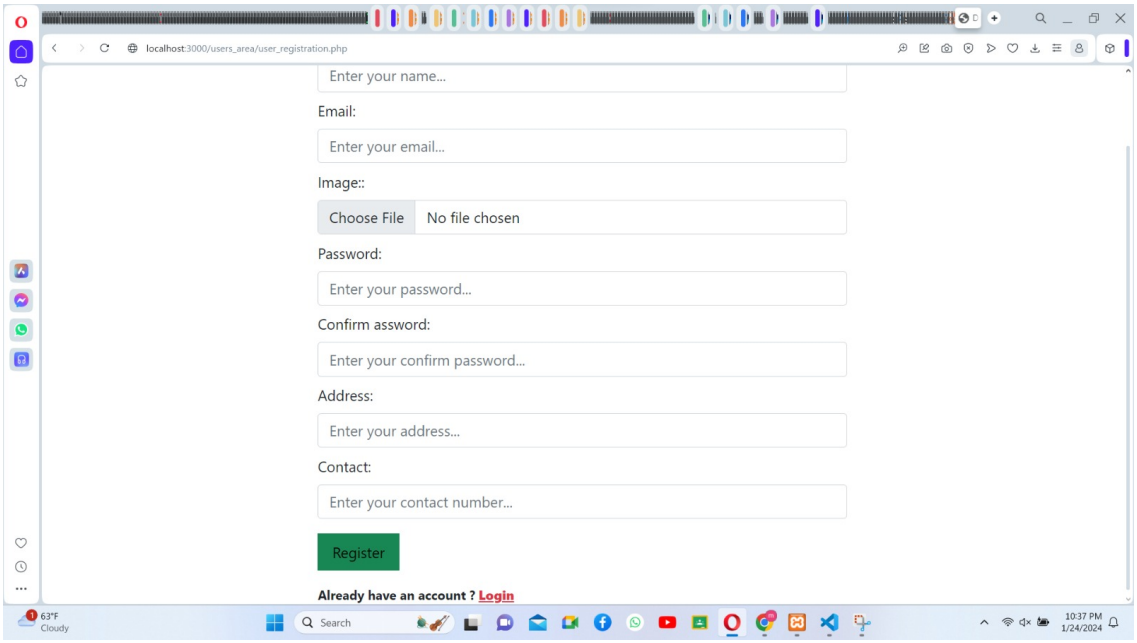
User Profile:



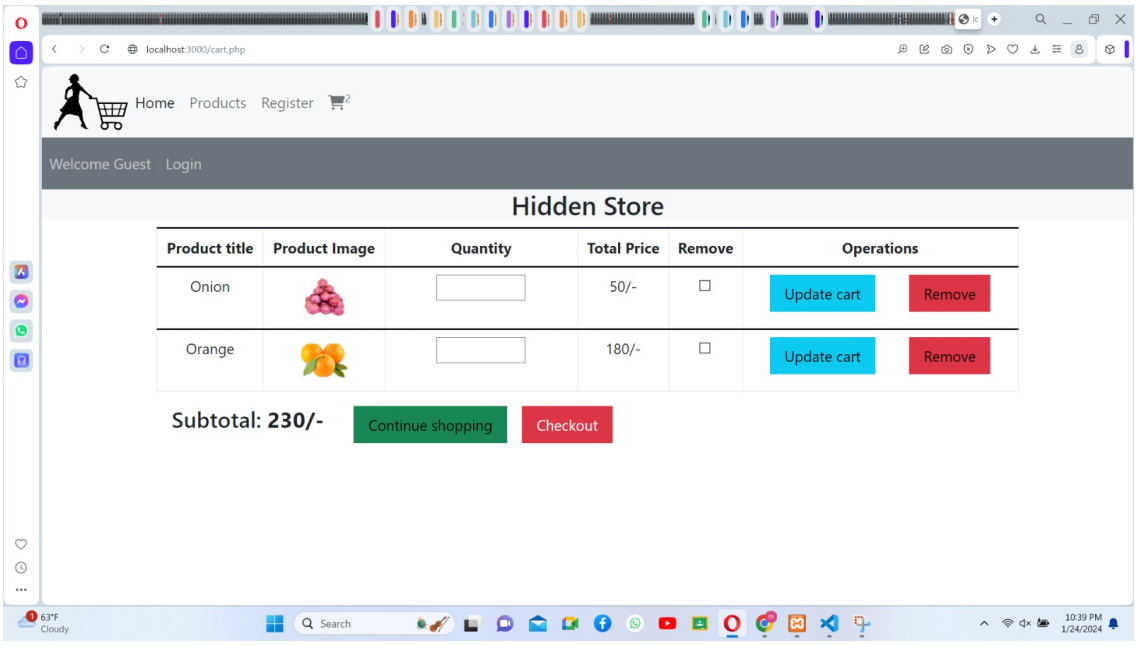
User registration:



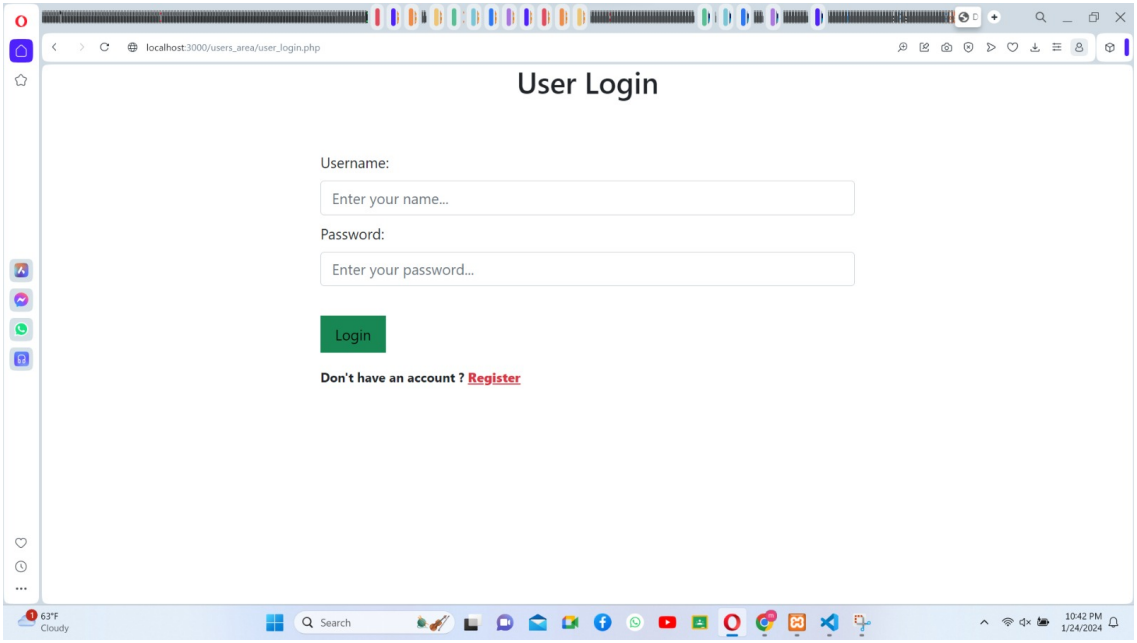
Registration bottom part:



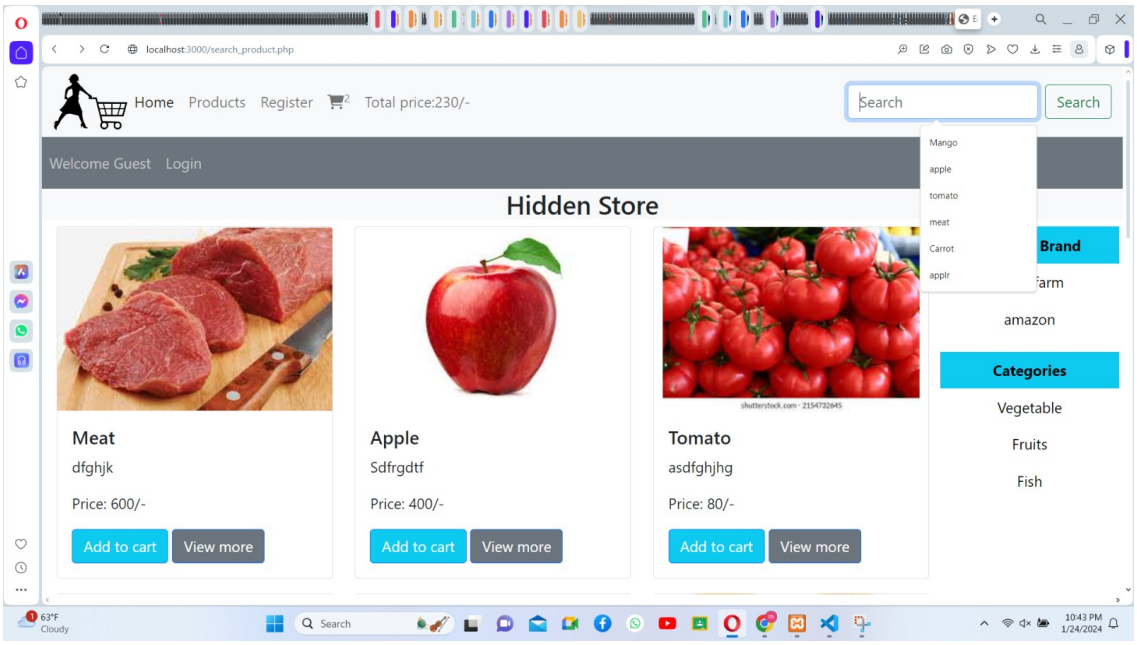
Cart:



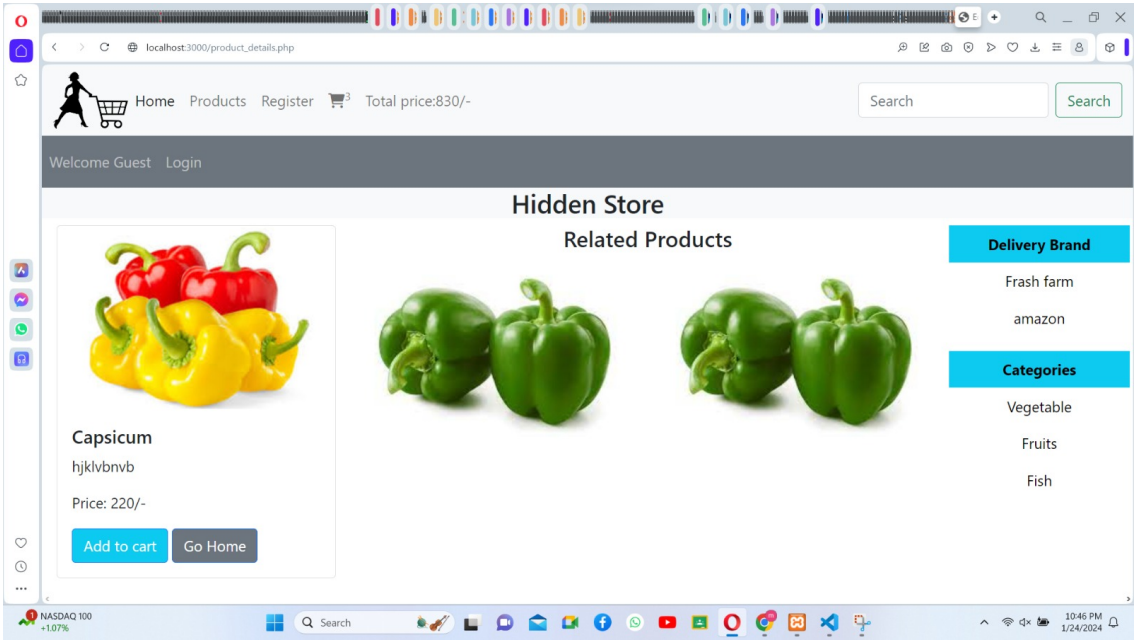
User Login:



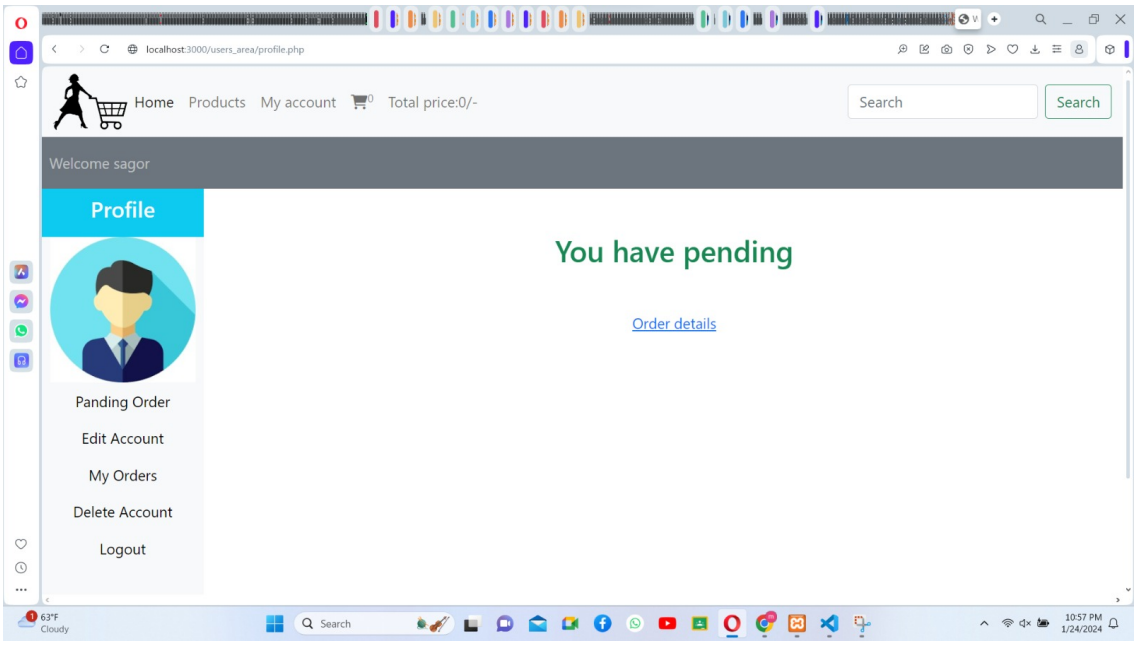
Search var:



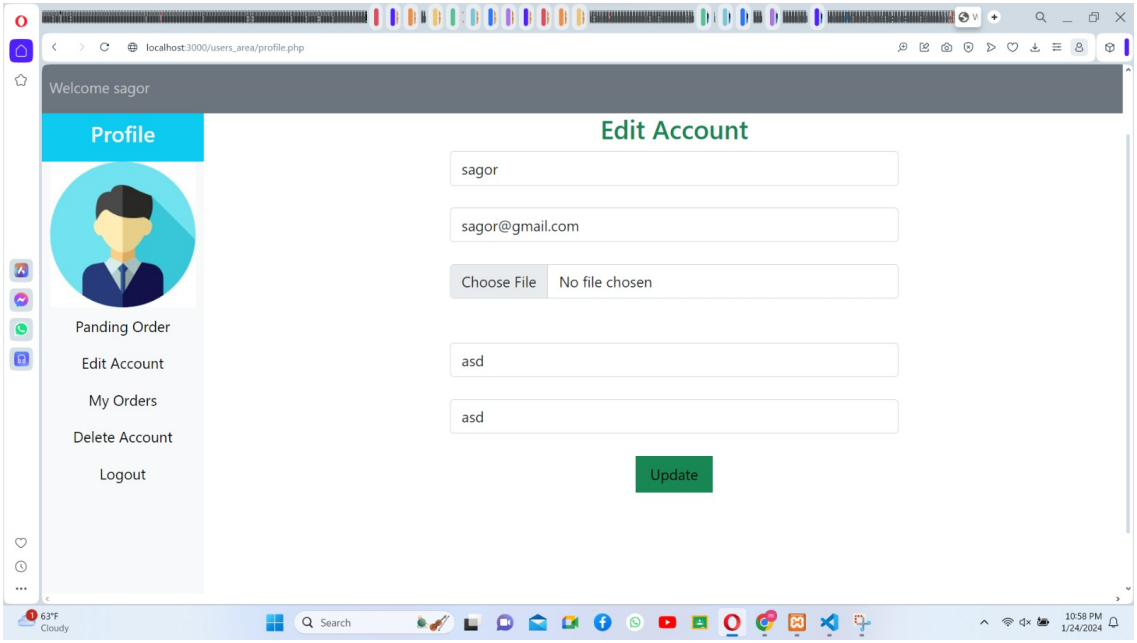
View more:



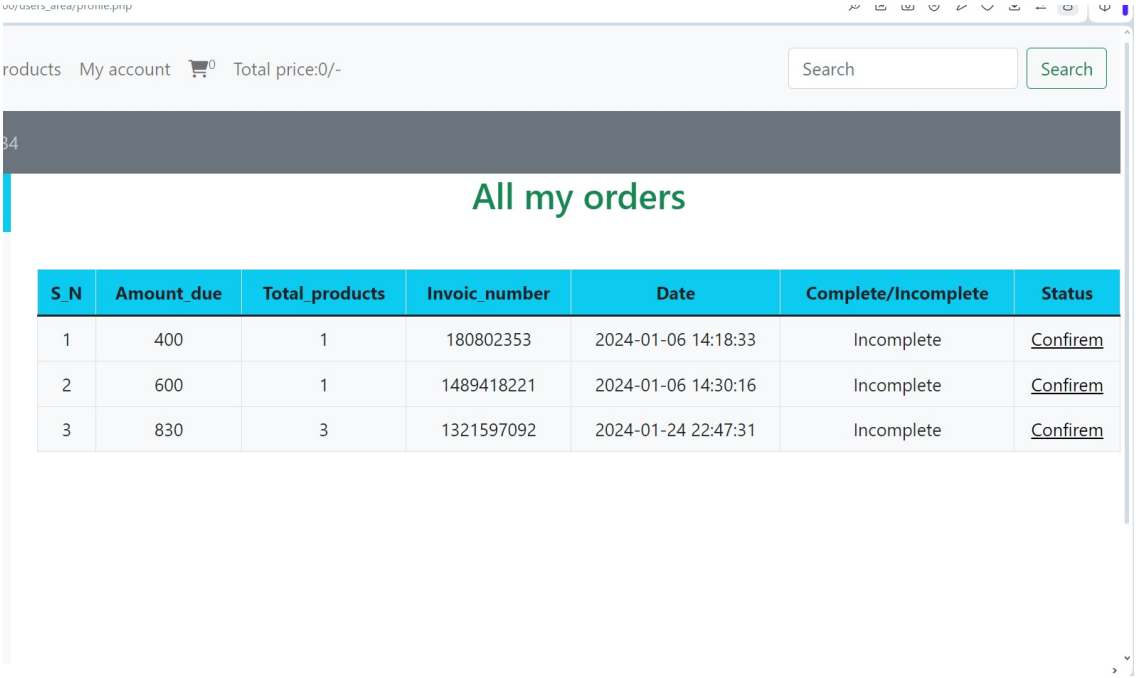
user profile:



Edit user account:

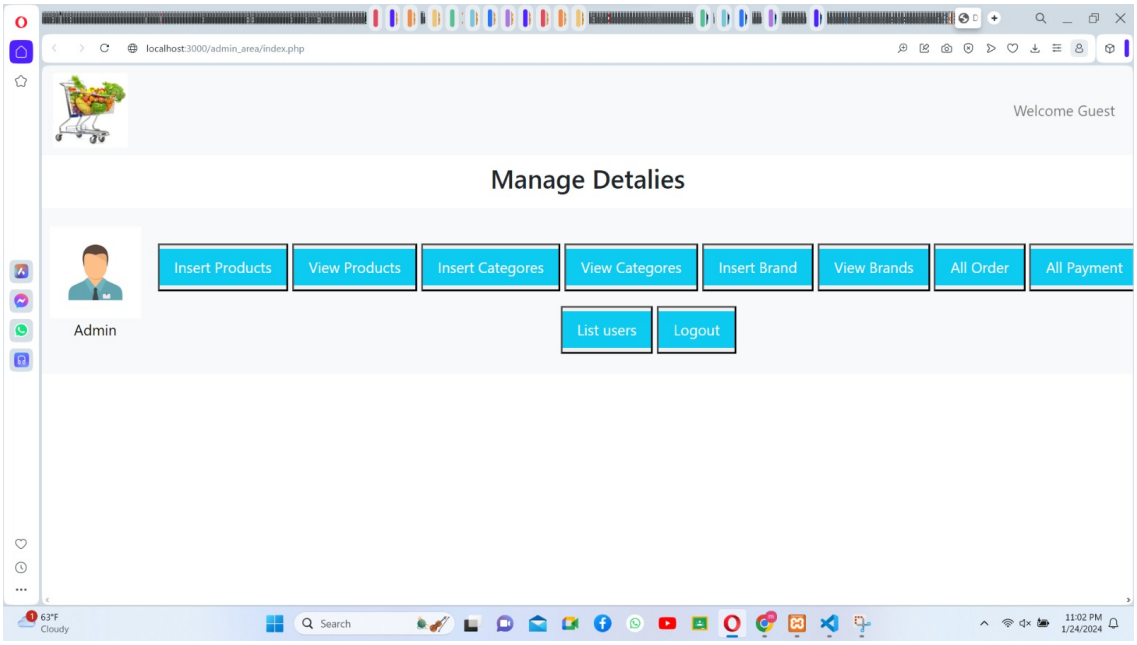


user all orders:

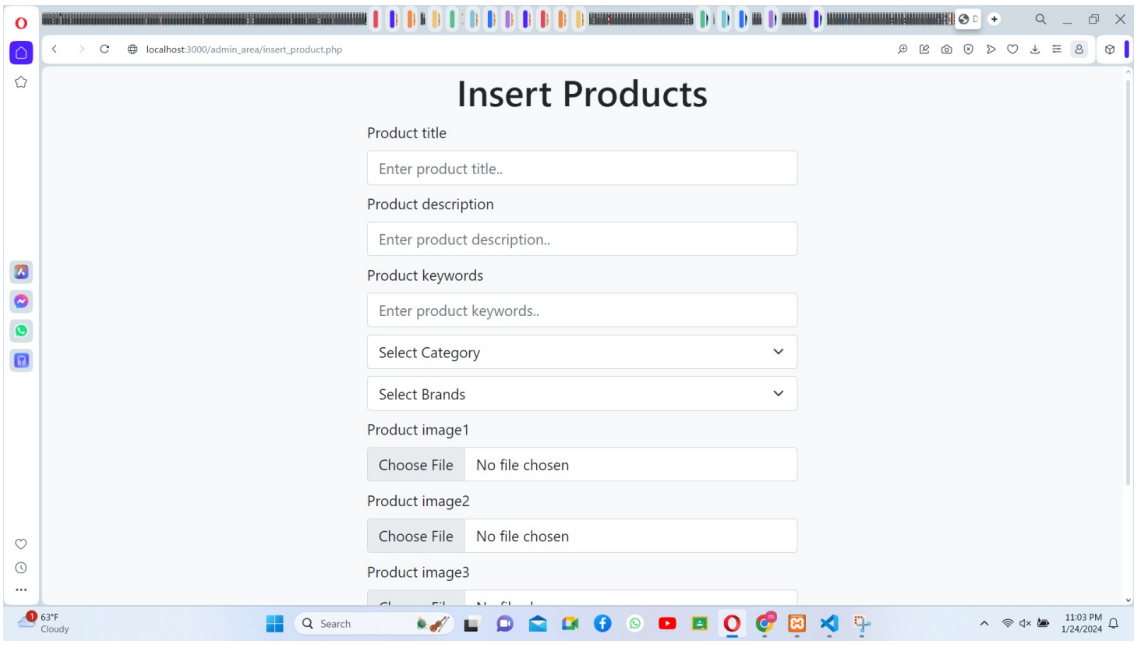


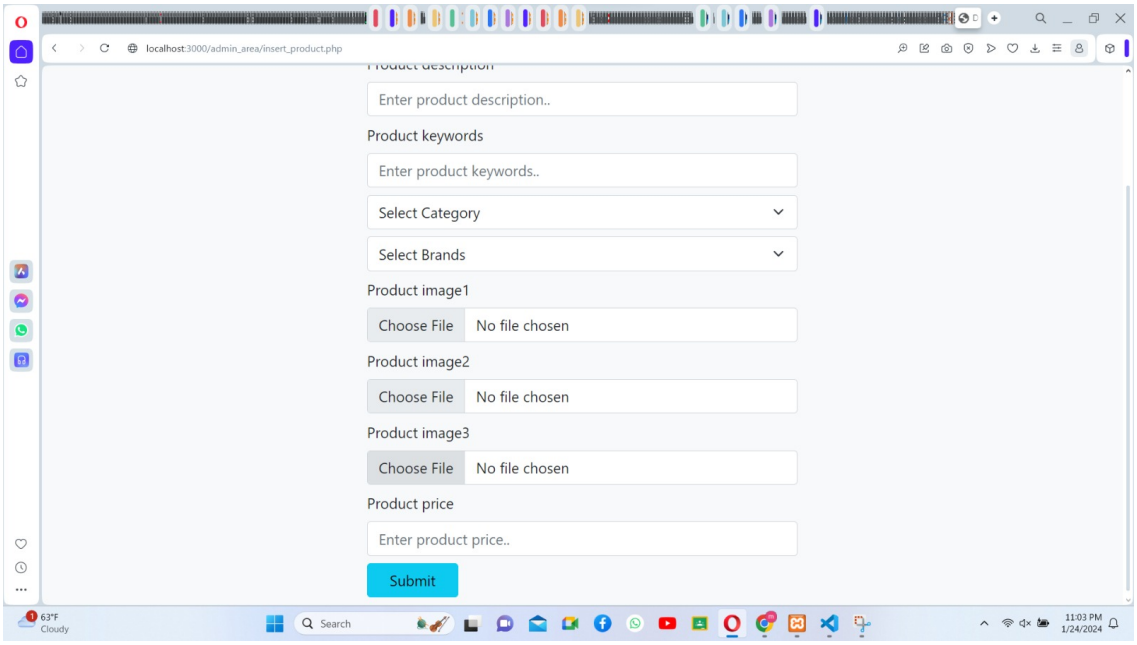
S_N	Amount_due	Total_products	Invoic_number	Date	Complete/Incomplete	Status
1	400	1	180802353	2024-01-06 14:18:33	Incomplete	Confirem
2	600	1	1489418221	2024-01-06 14:30:16	Incomplete	Confirem
3	830	3	1321597092	2024-01-24 22:47:31	Incomplete	Confirem

Admin page:



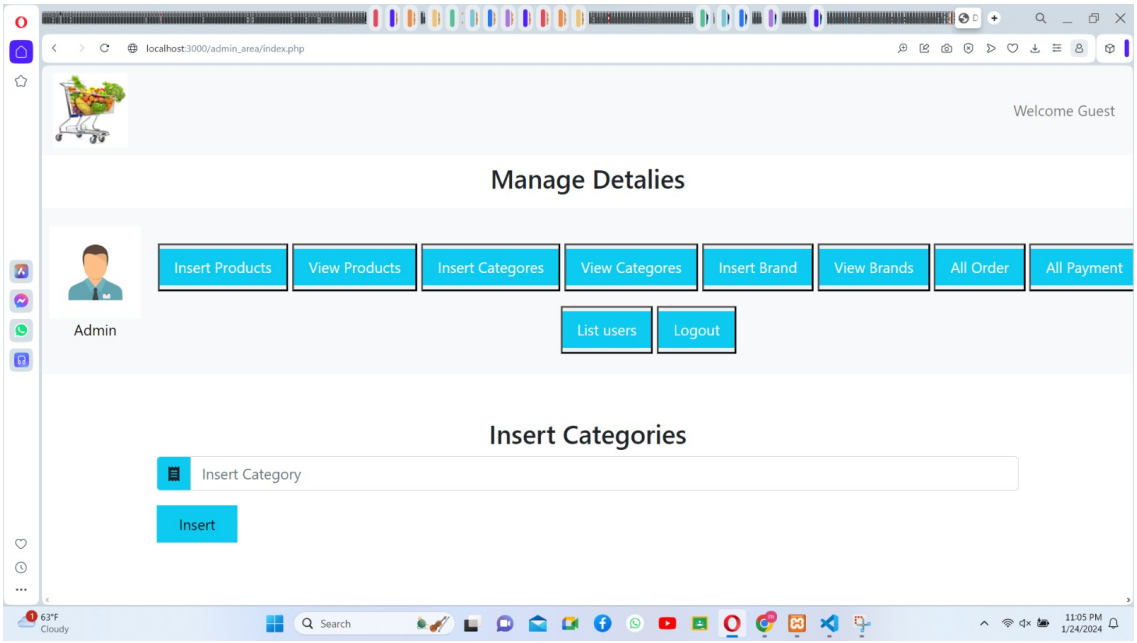
Insert products:



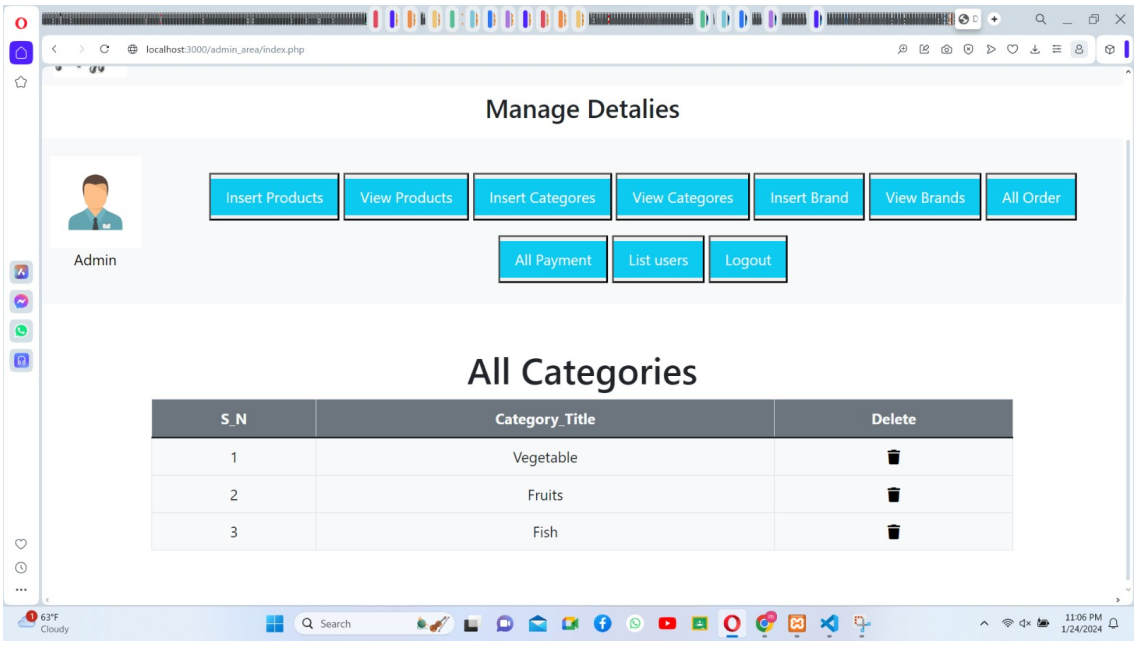


Admin view all products:

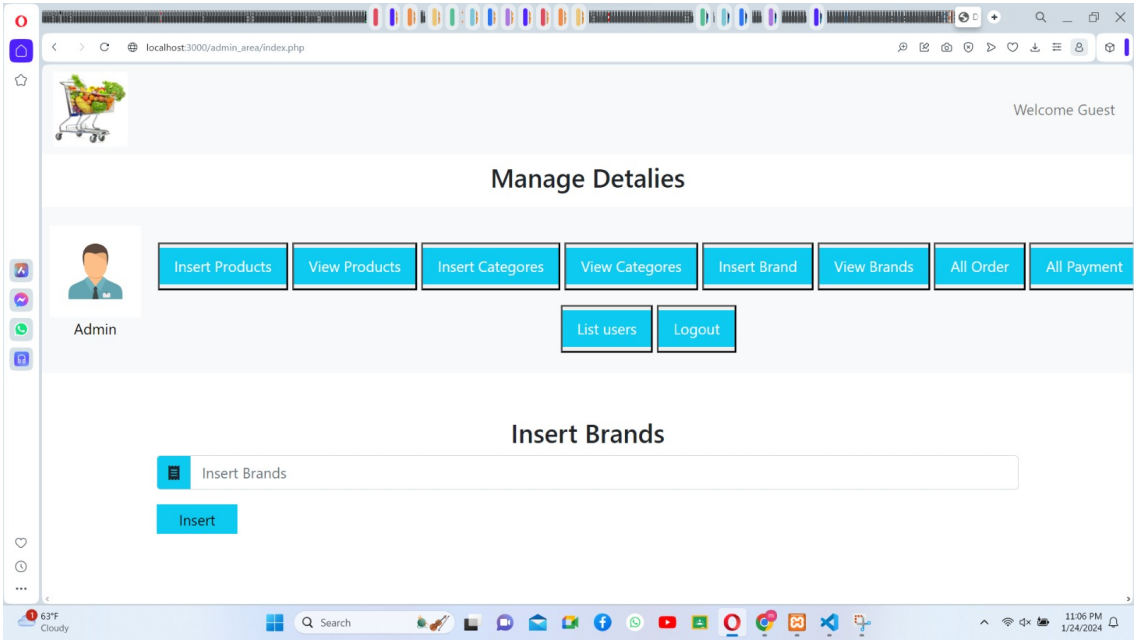
Insert Category:



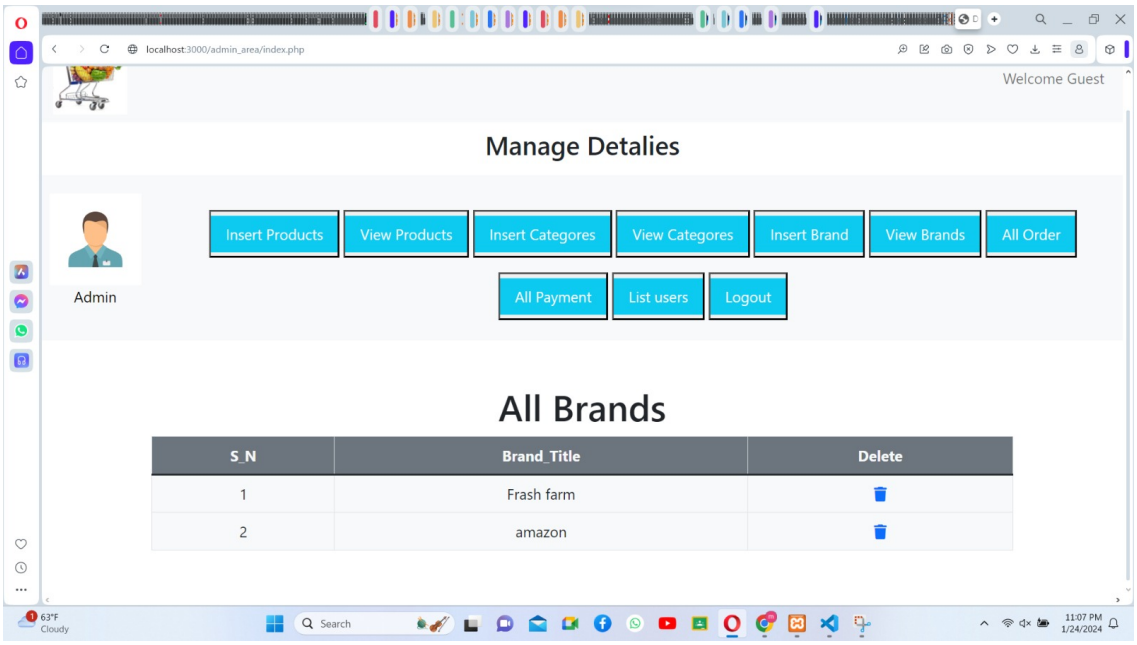
View all category:



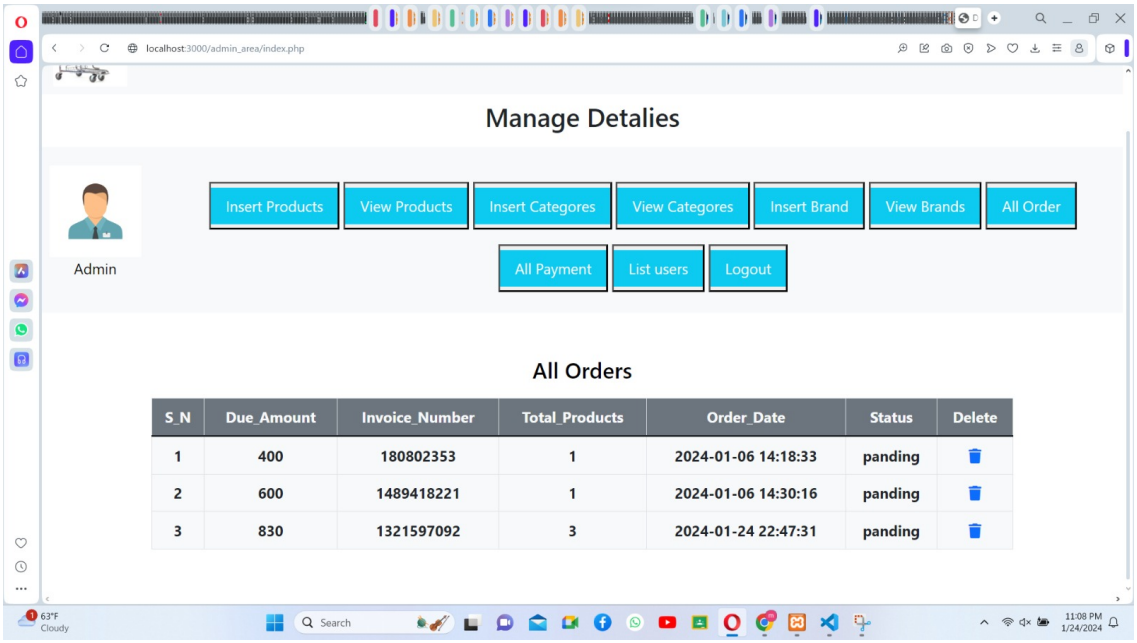
Insert Brand:



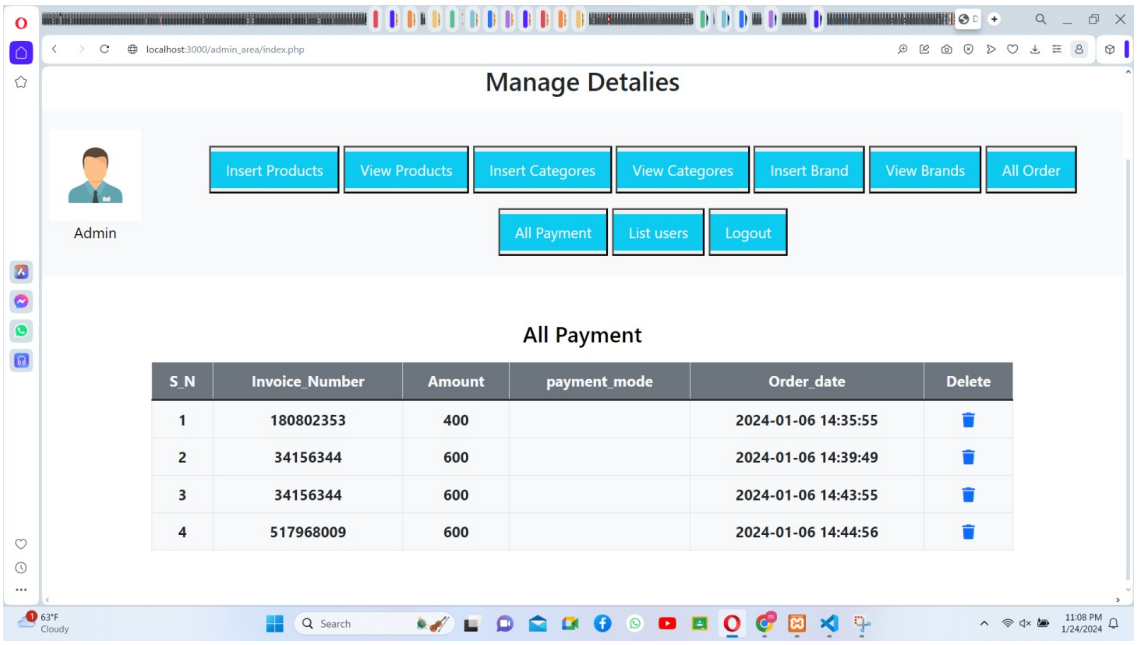
View all brands:



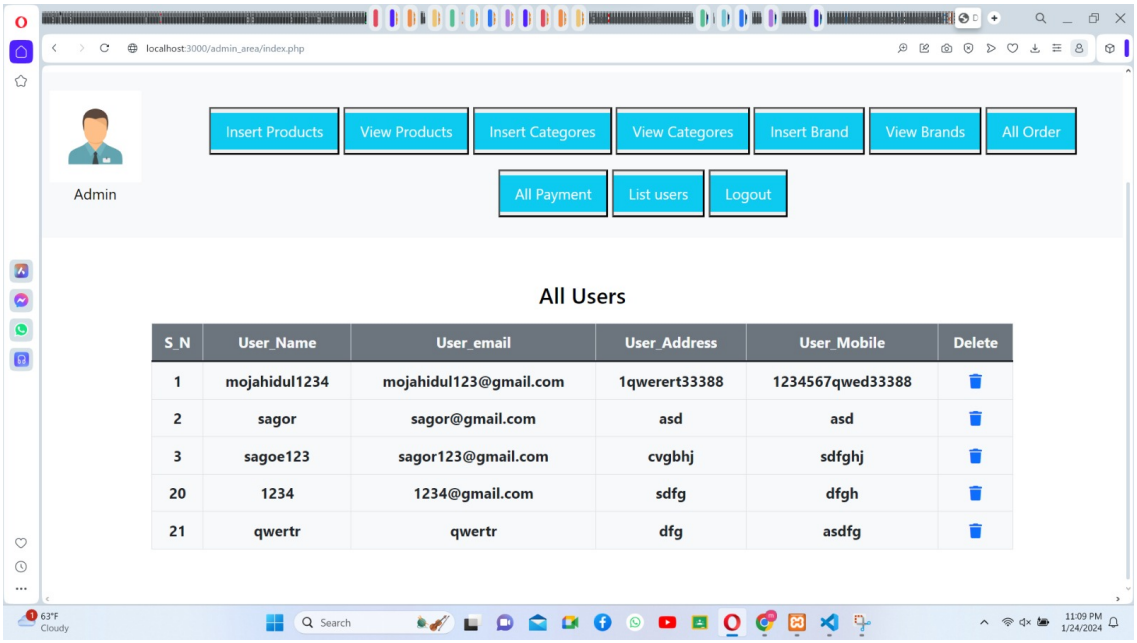
All orders:



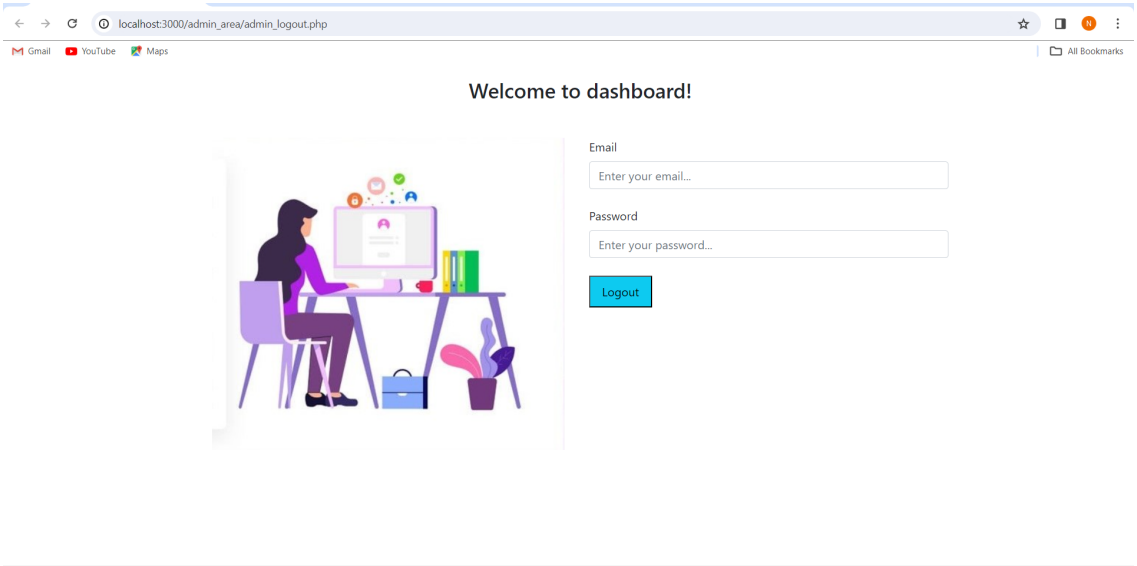
All payment:



All users:



Admin Logout:


localhost:3000/admin_area/admin_logout.php
Gmail
YouTube
Maps
All Bookmarks
Welcome to dashboard!
Email
Enter your email...
Password
Enter your password...
Logout

Testing and sustainability plan

6.1 Requirements/specifications-based system level test cases

An example table for Requirements/specifications-based system level test cases is given below:

Requirement ID	**Requirement Statement**	**Must/ Want**	**Comment**
R-Up-01	User can't upload image with invalid extension	Must	N/A
R-Up-02	Show error message if user upload image with invalid extension	Want	N/A
R-Gen-01	User can't generate caption without uploading image	Must	N/A
R- Gen-02	Show error message if user click generate caption option without uploading image	Want	N/A
R-Clean-01	For training the model with dataset, cleaning needs to be performed to the text dataset.	Must	N/A

Table 6.1: Requirements/specifications-based system level test cases (01)

Project Name	Software Development for Web Apps Sessional					
Module Name	Upload					
Created By	Name 1					
Reviewed By	Name 1					
Date of Creation	15-02-21					
Date of Review	15-02-21					
Test Case ID	**Scenario**	**Steps**	**Test Data**	**Expected Result**	**Actual Result**	**Status**
TC-UP-01	Check upload image option with invalid file extension.	1. Go to the user interface 2.Click 'choose file' option 3.Upload file from PC's directory	File: new.pdf	Redirect to the initial state of user interface	Redirect to the initial state of user interface	Passed
TC-UP-02	-	-	-	-	-	Passed

Table 6.2: Requirements/specifications-based system level test cases (02)

Project Name	Software Development for Web Apps Sessional					
Module Name	Text Cleaning					
Created By	Name 1					
Reviewed By	Name 1					
Date of Creation	15-02-21					
Date of Review	15-02-21					
Test Case ID	**Scenario**	**Steps**	**Test Data**	**Expected Result**	**Actual Result**	**Status**
TC-CL-01	-	-	-	-	-	Passed
TC-CL-02	-	-	-	-	-	Passed

Table 6.3: Requirements/specifications-based system level test cases (03)

Project Name	Software Development for Web Apps Sessional					
Module Name	Generate Caption					
Created By	Name 1					
Reviewed By	Name 1					
Date of Creation	15-02-21					
Date of Review	15-02-21					
Test Case ID	**Scenario**	**Steps**	**Test Data**	**Expected Result**	**Actual Result**	**Status**
TC-GEN-01	-	-	-	-	-	Passed
TC-GEN-02	-	-	-	-	-	Passed

Table 6.4: Requirements/specifications-based system level test cases (04)

6.2 Traceability of test cases to use cases

Test Case ID	Requirement ID				
	R-Up-01	R-Up-02	R-Gen-01	R- Gen-02	R-Clean-01
TC-UP-01	✓				
TC-UP-02	✓				
TC-GEN-01	✓				
TC-GEN-02	✓				
TC-CL-01	✓				
TC-CL-02	✓				

Table 6.5: Traceability of test cases to use cases

6.3 Techniques used for test generation

To ensure comprehensive coverage and robust testing, a variety of techniques were employed during test case generation:

Boundary Testing: Test cases were designed to explore the limits and boundaries of system functionalities, such as maximum and minimum input values, to ensure that the system behaves correctly under extreme conditions.

Positive and Negative Scenario Testing: Positive scenarios were tested to verify that the system functions as intended under normal circumstances. Negative scenarios, on the other hand, were explored to identify how the system handles unexpected or invalid inputs, ensuring error messages and fallback mechanisms were appropriately implemented.

Exploratory Testing: This technique involved dynamic exploration of the system to discover any unforeseen issues or potential vulnerabilities. Testers interacted with the system in an ad-hoc manner, simulating real-world usage patterns.

6.4 Assessment of the goodness of your test suite

The test suite demonstrated its effectiveness by ensuring that critical requirements were met. All test cases passed, indicating a high degree of test coverage and robustness in the system. The assessment of the test suite's effectiveness was conducted through rigorous test execution, and the following observations were made:

Test Coverage: The test suite achieved high coverage of critical functionalities, ensuring that all specified requirements were thoroughly tested.

Error Identification: The test suite successfully identified and flagged errors, allowing for prompt rectification of issues during the development phase.

Reproducibility: Test cases were designed to be easily reproducible, enabling consistent verification of system behavior and facilitating future testing efforts.

Consistency: The test suite maintained consistency in its approach, ensuring that similar functionalities were tested using uniform methods and criteria.

6.5 Sustainability Plan

6.5.1 Scalability

The application's design ensures scalability to accommodate a growing number of users and products. The architecture allows for the seamless addition of server resources to handle increased loads. The sustainability plan focuses on the scalability of the system to accommodate future growth:

Vertical Scaling: The system architecture supports vertical scaling by adding more resources to the existing server, ensuring that the application can handle increased load by leveraging additional CPU, memory, or storage capacity.

Horizontal Scaling: The application is designed to scale horizontally, allowing for the deployment of multiple instances across different servers. Load balancing mechanisms are in place to distribute incoming traffic evenly and ensure optimal performance.

6.5.2 Flexibility / Customization

The system offers flexibility and customization options, particularly in product details. Utilizing Bootstrap allows for easy adaptation of the user interface to different devices and screen sizes. Flexibility and customization are key aspects of the sustainability plan:

User Interface Customization: The utilization of Bootstrap as the framework enhances flexibility in adapting the user interface to different devices and screen sizes. This ensures a seamless user experience across a variety of platforms.

Product Details Customization: The system allows for easy customization of product details, facilitating the addition or modification of information related to vegetables and fruits. This ensures adaptability to changing business requirements and evolving product catalogs.

Acknowledgement

Bismillahir-Rahmanir-Rahim, Foremost, we want to show gratefulness to our Almighty, granting us the strength and patience to see through the completion of this project successfully. Without His endless blessings, it would have not been possible to complete such a task within the given time. We would also like to show our sincere gratitude to our project supervisor and course teacher Rabeya Sultana for her endless support and supervision. Her support throughout the project had helped us better understand the task we are to perform, and the result we are to produce. He helped us look into matters that would have been overlooked otherwise.

We recall our teachers, friends, and all others who had inspired and helped us throughout the completion of the project.

——- 2024

References