**2n1 Café Corp: Branding, POS and Ordering using Kiosk Application**

An

Application Development Project

Presented to the Faculty of

**Mindoro State University Calapan City Campus**

Masipit, Calapan City

Oriental Mindoro

In Partial Fulfillment

of the Requirements for the Degree

Bachelor of Science in Information Technology

by:

**Dilay, Waren D.**

**Bahia, Rhay Bien D.**

**Cataquis, Hanz Diane D.**

*October 7, 2024*

**TABLE OF CONTENTS**

**LIST OF FIGURES**

**LIST OF TABLES**

**LIST OF APPENDICES**

**CHAPTER I**

**INTRODUCTION**

**Project Context**

One way to gain customers’ trust, loyalty, and satisfaction is to focus on providing services and engaging with people. The rapid development of technologies are increasingly utilized to enhance operational efficiency and deliver superior services (Kumari et al., 2024)(Wei & Simay, 2023)just like Kiosk Application (Naidu et al., 2024). Self-ordering is an idea (Yee et al., 2024). A self-ordering kiosk technology allows users to get information or conclude their purchases without human engagement. A kiosk system is an interactive self-ordering device that allows customers to browse and order. A kiosk system allows them to get the products they need while saving time. They are commonly used to boost customer satisfaction and efficiency. A kiosk system allows customers to make purchases quickly and easily. More people are turning to self-ordering devices, such as kiosks, enabling easier and faster purchasing (Naidu et al., 2024). This study examines the implementation and effectiveness of self-ordering kiosk systems in large restaurant or café (Naidu et al., 2024). These technologies are becoming more common in the lodging, food and beverage, travel, and transportation industries.

Despite some criticism from stakeholders, these technologies are becoming more common across numerous industries (Kumari et al., 2024) (Wei & Simay, 2023). Various sectors, including the food and beverage industry, are integrating technology into their operations to enhance corporate growth. Technology-enabled business operations, such self-ordering kiosks, iPads, and QR code menus, are becoming increasingly popular in stores, hotels, airports, and restaurants. The book menu eliminates the inefficiency of manual ordering while also minimizing paper waste.

Many restaurants offer QR codes for patrons to request menus directly from their tables. The purpose of this study is to investigate the influence of self-ordering with QR codes in the scientific community (Susanti et al., 2024). Implementing a Kiosk Application has enormous possibilities for improving the customers’ experience, simplifying operations, and offering cost saving testing. To maintain the system's continuous efficacy and success in addressing the shifting demands of consumers, it is vital to handle any identified challenges and aggressively explore opportunities for development(Naidu et al., 2024).

**Objectives**

**General Objective:** To develop a kiosk-based ordering application that enhances the ordering process for customers and provides real-time sales reporting for the admin.

**Specific Objectives:**

1. To streamline the customer ordering process through a user-friendly kiosk interface.
2. To enable efficient tracking of orders and sales management.
3. To simplify scheduling of kiosk operation and maintenance.
4. To provide real-time sales reports for the admin.

**Scope and Limitations**

This project aims to develop a Kiosk Application to optimize the ordering process at the café, improving both customer experience and operational efficiency. The kiosk will enable customers to browse the menu, place orders independently, and streamline the order fulfillment process. Additionally, the system will provide the admin with real-time access to sales data, allowing for effective sales tracking.

Although the system has several of unique features, it also considers potential drawbacks such as technical issues and user resistance:

1. **Lack of Familiarity:** Some customers and employees may be unfamiliar with using kiosk technology, leading to hesitation or frustration during initial interactions.
2. **Training Requirements**: Staff may need training to assist customers with using the kiosk system effectively and ensuring smooth, consistent operation.

**Definition of Terms**

**Kiosk Application** – a self-service software system that enables customers to browse the menu, and place orders.

**Sales Report** – a comprehensive summary of sales transactions processed by the kiosk application.

**Real-time monitoring** - the continuous tracking and reporting of key data such as sales transactions, employee attendance, and system performance as it happens.

**Queue Management -** organizes and manages the sequence of customer orders.

**User Interface (UI) -** allowing customers and admins to easily navigate through the ordering, updating, and other features.

**Admin Tools –** a centralized interface for administrators to monitor sales, manage products and other administrative data.

**CHAPTER II**

**REQUIREMENTS SPECIFICATION**

**Hardware and Software Requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| **Software** | **Type** | **Minimum Specification** | **Recommended Specification** |
| Node.js | Backend Framework | Version 12.0 or higher | Version 12.0  or higher |
| Express.js | Web Framework | Web Framework | Web Framework |

**Functional Requirements**

1. **Real-Time Sales Report/Monitoring**

* For easy identification of what are the products that are mostly purchased.

1. **Placing Orders**

* Placing orders based on quantity. This would involve order management that processes a first-come, first-serve basis.
* Allow customers to customize their orders (e.g., size, ingredients, and other specific instructions).

1. **Queue Management**

* First Order, First Serve: You want to implement a queue system for orders, where the first person to place an order is the first to be served.

1. **User Interface**

* The system must offer a clean, intuitive interface for easy navigation of the users.

1. **User Management**

* The system must have login functionality for owner, admin/employee, and customers. It should also have different access levels to functionalities based on the user’s role.

1. **Inventory Management**

* The system must offer stock tracking to monitor inventory levels and provide alert for low stock items.

**Non-Functional Requirements**

1. **Operational Requirement**

**System Uptime:** The kiosk application must be operational to allow continuous monitoring of employee attendance, real-time sales data without any downtime.

1. **Performance Requirement**

**Order Display:** Customers order must be displayed for the order placement via the kiosk.

**Automated Sales Report:** Real-time sales reporting should be accurate and complete, providing the admin with up-to-date business insights.

1. **Security Requirement**

**User Authentication:** Access to the admin dashboard, HR management tools, and sensitive reports should be protected by strong authentication methods (e.g., passwords, two-factor authentication).

**Database Security**: All stored data (e.g., sales reports, etc.) must be accessible only to authorized users.

**Chapter III**

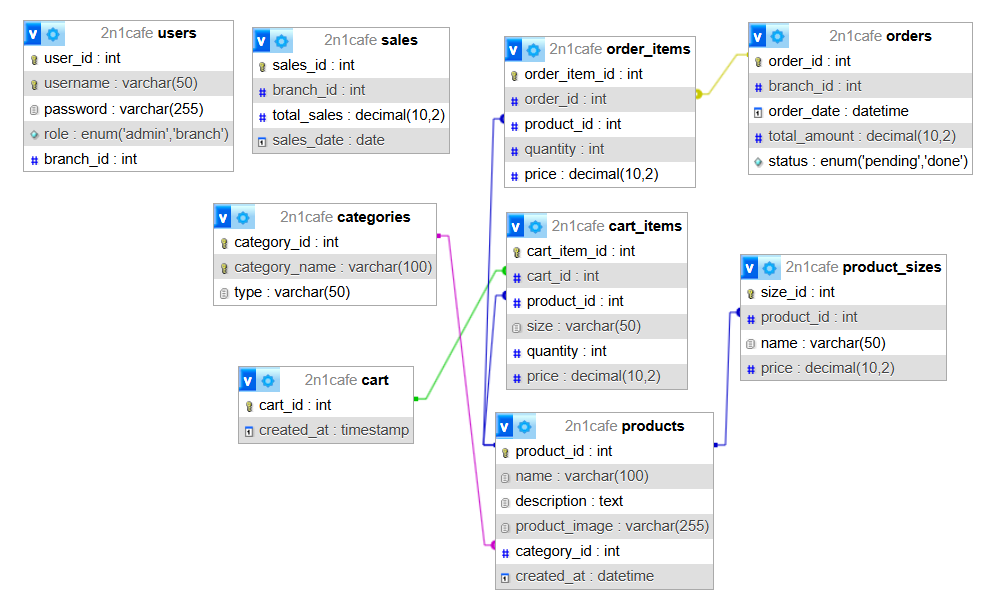
**Design and Development Methodologies**

**System Design**

2n1 Café Corp: Branding, POS, and Ordering using Kiosk Application consists of a three-tier architecture: the presentation layer (frontend), business logic layer (backend), and the data layer (database). The frontend includes a user-friendly kiosk interface that allows customers to browse the menu and customize orders. For administrators, a dashboard offers real-time sales monitoring, and interfaces for customizing branches, products and kiosk.

The backend is powered by a Node.js server that manages core operations such as order processing, real-time sales reporting, and inventory tracking. To ensure secure data transmission between the kiosk and backend, communication is encrypted. Authentication and authorization measures are in place to protect sensitive information and control access to the admin dashboard based on user roles.

In the data layer, a relational database like MySQL is used to store key data, including sales transactions and product inventory. This setup supports real-time sales analysis and inventory management, ensuring smooth order fulfillment and efficient reporting. This holistic design ensures smooth operations, efficient management, and a seamless customer experience while supporting future scalability.

**Database Design**

**Figure 1: Database System Design**

Figure 1 shows the database designed for managing 2n1cafe’s operations, including users, products, sales, orders, and carts. Tracking user roles and branch in *users* table, branch sales and total sales in the *sales* table, and customer orders in the *orders* and *order\_items* table. Products are stored in the *products* table, categorized by the *categories* table and detailed with size options in the *product\_sizes* table. Shopping carts and their items are managed through the cart and *cart\_items* tables.

**Table 1: Fields of Users**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Data Type | Default | Description |
| user\_id | int | *Not NULL* | user's id (PK) (AI) |
| username | varchar(50) | *Not NULL* | user’s own username |
| password | varchar(255) | *Not NULL* | user's password |
| role | enum(‘admin’,’branch’) | *NULL* | user's role |
| branch\_id | int | *NULL* | branch's id |

Table 1 shows the field of users or users table. This table stores user’s information, including user\_id as the unique primary key (auto incremented), username and password for login credentials, role, and also the branch\_id.

**Table 2: Field of Sales**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Data Type | Default | Description |
| sales\_id | int | *Not NULL* | sales’ id (PK) (AI) |
| branch\_id | int | *Not NULL* | branch's id |
| total\_sales | decimal (10,2) | *Not NULL* | number of total sales |
| sales\_date | date | *Not NULL* | date when transaction occurred |

Table 2 shows the field of sales or sales table. This table stores sales records, including sales\_id as primary key (auto incremented), branch\_id, total\_sales records the total amount of sales, and sales\_date indicates the date when transaction occurred.

**Table 3: Field of Product Sizes**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Data Type | Default | Description |
| size\_id | int | *Not NULL* | size's id (PK) (AI) |
| product\_id | int | *Not NULL* | product's id |
| name | varchar(50) | *NULL* | name and size of product |
| price | decimal(10,2) | *Not NULL* | product's price |

Table 3 shows the field of product sizes or product sizes table. This table manages the product sizes, size\_id as the primary key (PK)auto-incremented (AI), product\_id, name specifies the name and size of product, and price for the price of the product’s size.

**Table 4: Field of Products**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Data Type | Default | Description |
| product\_id | int | *Not NULL* | product’s id (PK) (AI) |
| name | varchar(100) | *Not NULL* | product's name |
| description | text | *NULL* | product's description |
| product\_image | varchar(255) | *Not NULL* | image of product |
| category\_id | int | *NULL* | id of product’s category |
| created\_at | datetime | *NULL* | default generated, current timestamp |

Table 4 shows the field of products or products table. This table stores product information, including product\_id as primary key (PK) and auto-incremented (AI)to uniquely identify each product, name for product’s name, description (optional), product\_image for the product’s image, category\_id, and created\_at when the product was added.

**Table 5: Field of Order Items**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Data Type | Default | Description |
| order\_item\_id | int | *Not NULL* | ordered item’s id (PK) (AI) |
| order\_id | int | *Not NULL* | order's name |
| product\_id | int | *Not NULL* | product's id |
| quantity | int | *Not NULL* | number of ordered items |
| price | int | *Not NULL* | product's price |

Table 5 shows the field of the order items. This table handles the ordered items, including order\_item\_id as the primary key (PK) and auto-incremented (AI), order\_id, product\_id, quantity for the number of items, and price represents the product’s price for the order.

**Table 6: Field of Orders**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Data Type | Default | Description |
| order\_id | int | *Not NULL* | order's id (PK) (AI) |
| branch\_id | int | *Not NULL* | branch's id |
| order\_date | datetime | *NULL* | default generated, current timestamp |
| total\_amount | decimal(10,2) | *Not NULL* | total amount of orders |
| status | enum(‘pending’, ‘done’) | *NULL* | order's status |

Table 6 shows the field of orders. This table stores order information, including order\_id as primary key (PK) and auto-incremented, branch\_id, order\_date when the order was placed, total\_amount for the value of order, and the status.

**Table 7: Field of Categories**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Data Type | Default | Description |
| category\_id | int | *Not NULL* | category's id (PK) (AI) |
| category\_name | varchar(100) | *NULL* | name of category |
| type | varchar(50) | *Not NULL* | type of category |

Table 7 shows the field of categories. This table defines product categories, including category\_id as the primary key(PK) and auto-incremented(AI) to uniquely identify each category. The category\_name optionally stores the name of the category, while type specifies the category type and is a required field.

**Table 8: Field of Cart Items**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Data Type | Default | Description |
| cart\_item\_id | int | *Not NULL* | cart item’s id (PK) (AI) |
| cart\_id | int | *Not NULL* | cart's id |
| product\_id | int | *Not NULL* | product's id |
| size | varchar(50) | *NULL* | product size |
| quantity | int | *Not NULL* | number of products |
| price | decimal(10,2) | *Not NULL* | price of products |

Table 8 shows the field of cart items. This tale shows the record items in shopping cart, including cart\_id as the primary key(PK) and auto-incremented(AI), cart\_id, product\_id, size for the size of the product (optionally), quantity for the number of products, and price for the total cost.

**Table 9: Field of Cart**

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Data Type | Default | Description |
| cart\_id | int | *Not NULL* | cart's id (PK) (AI) |
| created\_at | timestamp | *NULL* | default generated, current timestamp |

Table 9 shows the field of cart. This table stores shopping cart details, with cart\_id as the primary key (auto-incremented) to uniquely identify each cart. The created\_at field records the timestamp when the cart was created, defaulting to the current time if not specified.

**Architectural Diagram/ Block Diagram**

**PRESENTATION LAYER**

KIOSK INTERFACE ADMIN TOOLS

**BUSINESS LOGIC LAYER**

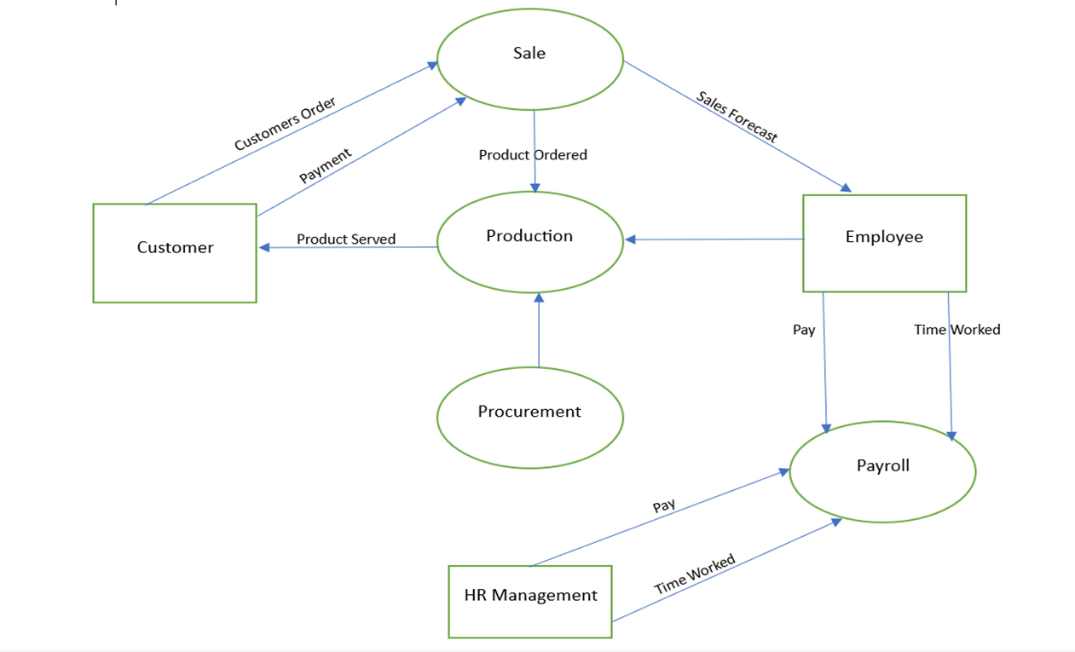
Node.js Server

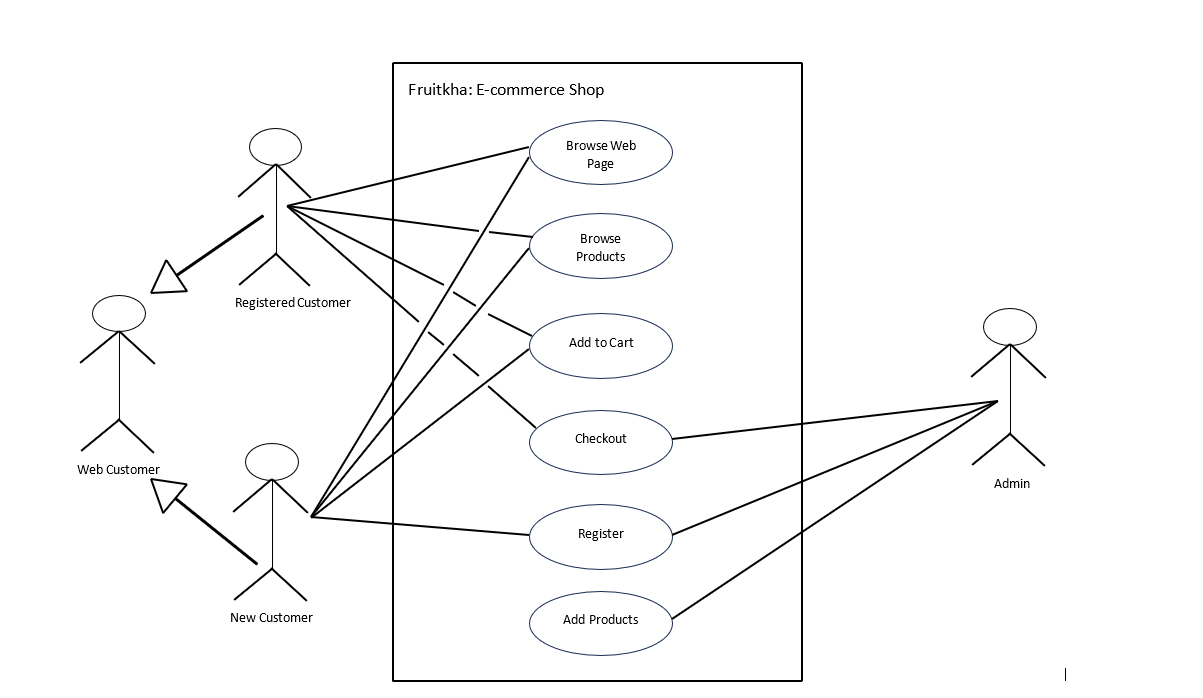
(Order Processing, POS, Sales)

**DATA LAYER**

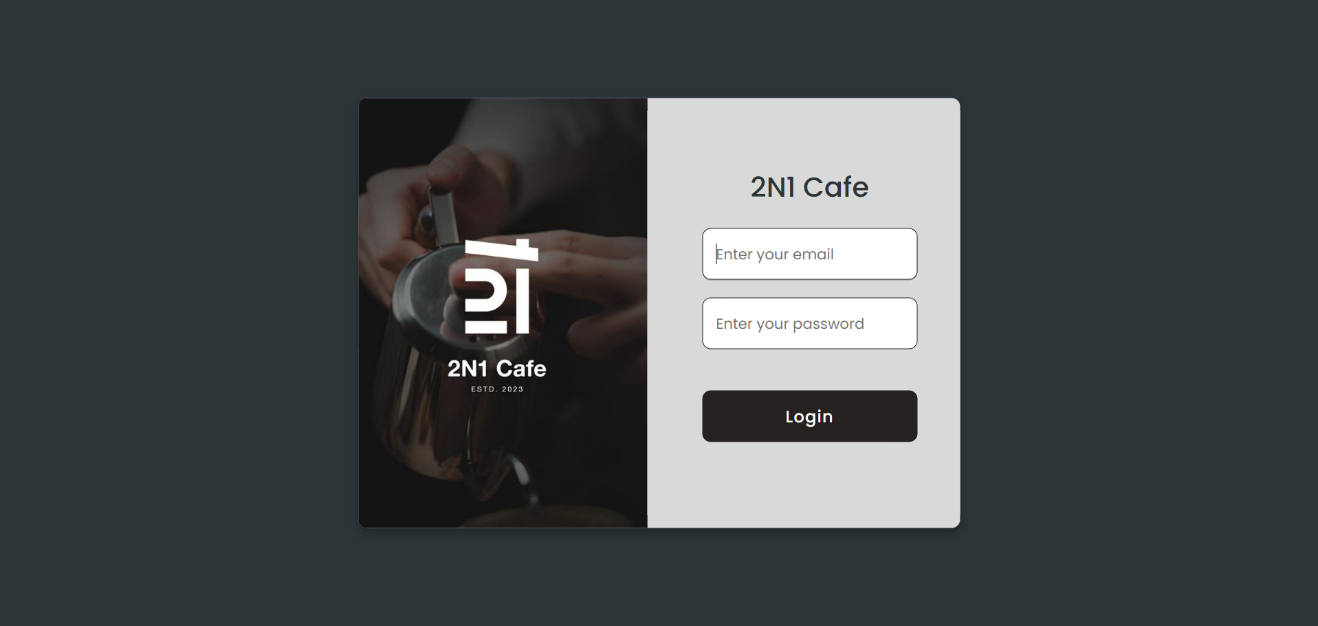
Database (MySQL)

(Branches, Sales, Products Data)

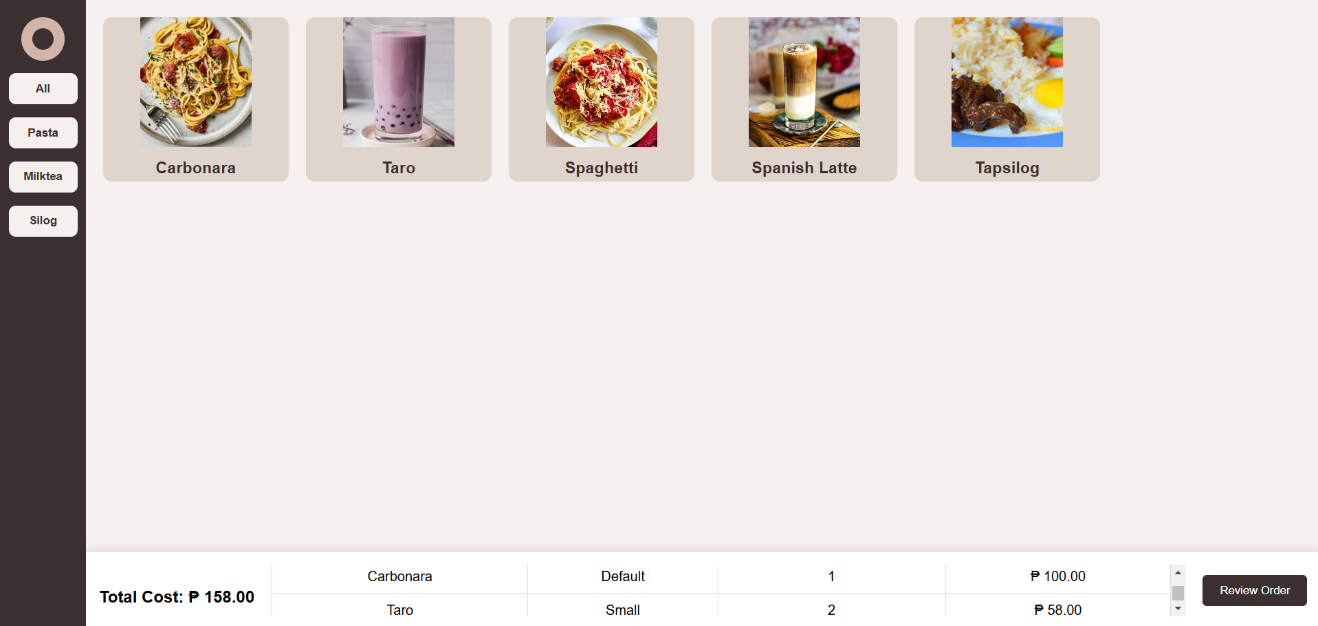
**DFD Level 0**

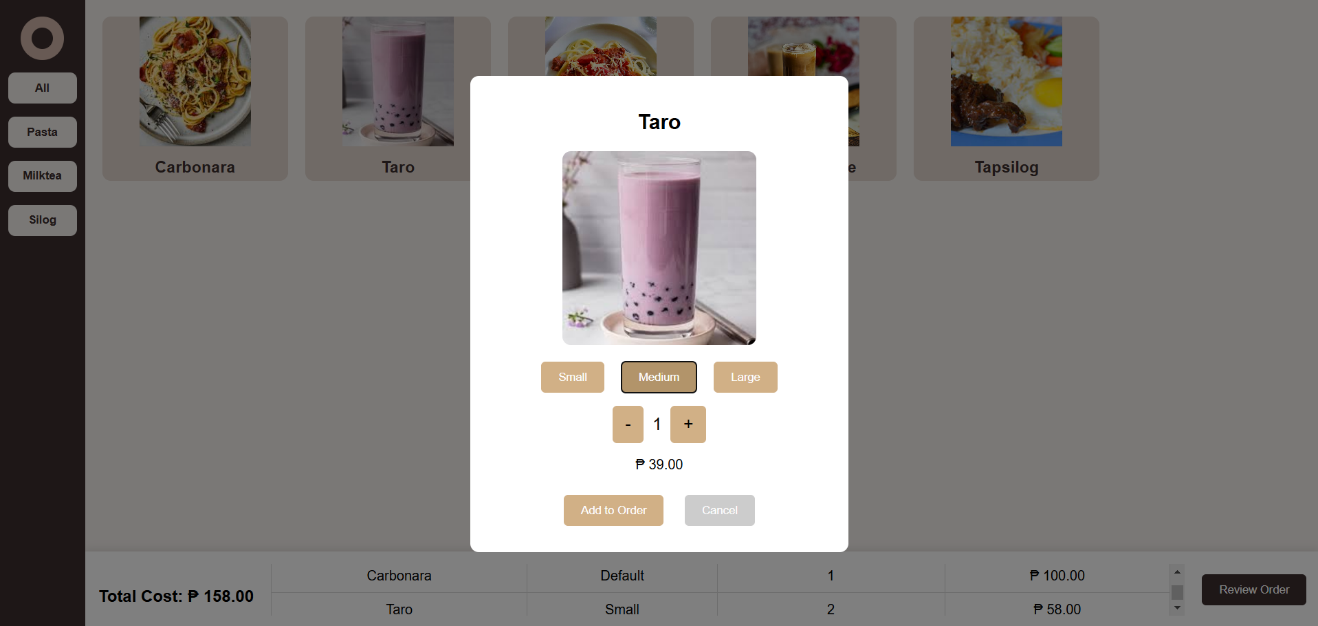
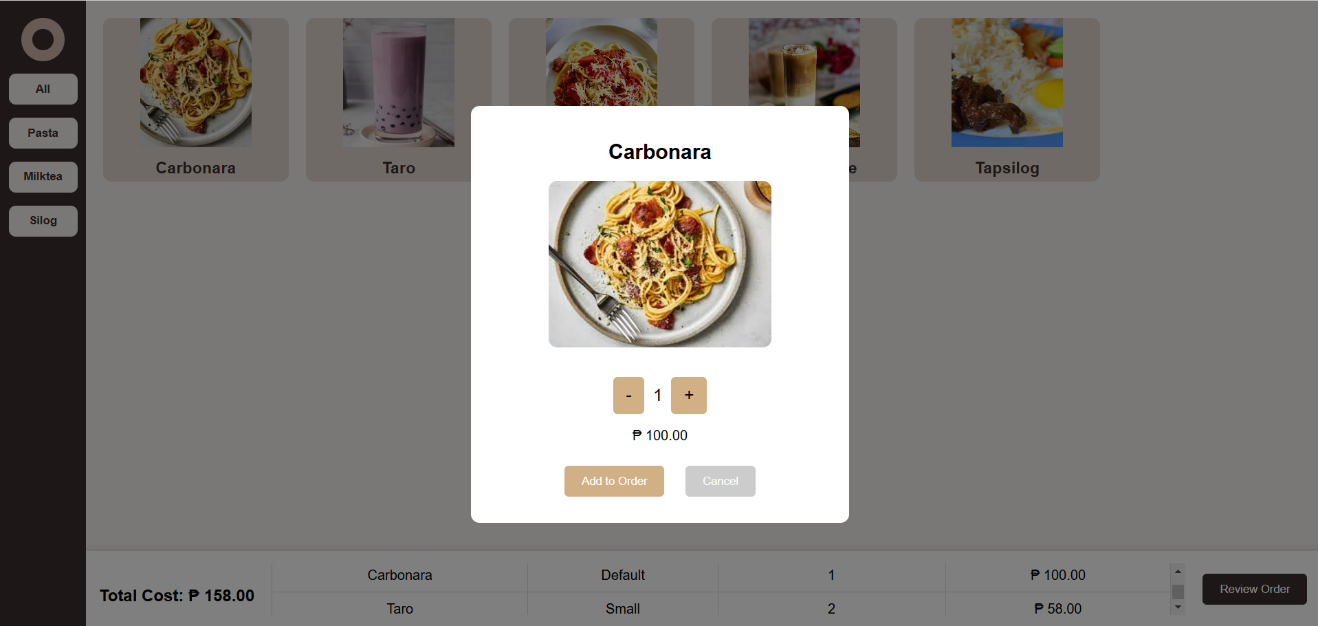
**UML Use-case Diagram**

**Sample Mock-up**

****

**Figure 2: Login Page**

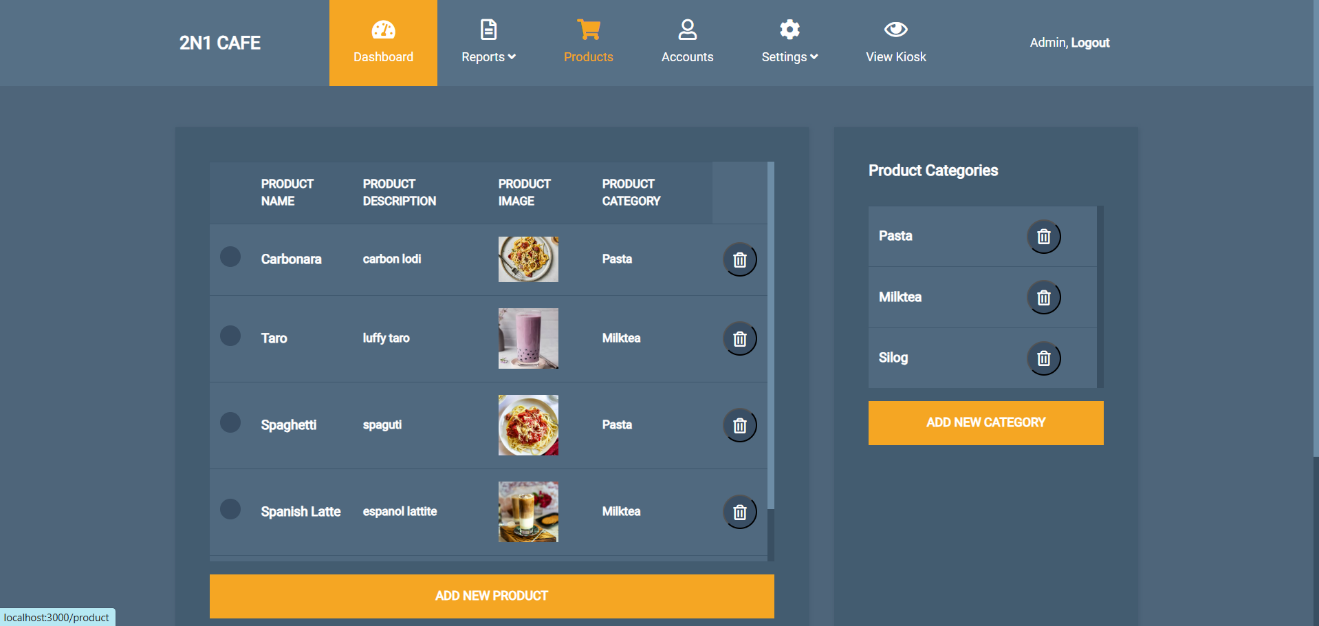
****Figure 2 shows the Log in page of 2n1 café, including the textbox where the user will input their email address and password. The Log in button, it is used to verify if the input is valid.

****

**Figure 3-4-5: Kiosk**

Figure 3 shows the Kiosk interface that displays the menu of items categorized by type with product images and name. Selected items are added to a cart at the bottom, listing the product name, size, quantity, and price. The total cost is calculated dynamically, with an option to review the

order before proceeding. Figure 4 and 5 shows a pop-up for selecting item also with the details, such as (Small, Medium, Large), quantity and price. User can add the ordered item to the cart or cancel the selection.

****

**Figure 6-7: Admin Side (Dashboard)**

Figure 6 and 7 shows the admin dashboard of 2n1 café. On the admin side, the interface allows the management of products, categories, and orders. Admins can add, update, or remove items from the menu, define product sizes and prices, and organize items into categories. They can also monitor and manage orders, including viewing order details, updating order statuses (e.g., from "pending" to "done"), and tracking sales data for specific branches or dates.

**Methodology**

This section describes the approach used to develop the 2n1 Cafe System, a solution designed to manage customer ordering and administrative tasks. It explains the process that involves gathering requirements, designing user-friendly interfaces, and developing key features such as menu display, cart management, order tracking, and sales reporting. The primary focus is to create a system that is efficient, accurate, and adaptable to the evolving needs of the café.

For developing the 2n1 café, the Agile Methodology is the best choice because it offers flexibility and iterative approach, which allow for continuous improvement and regular feedback from stakeholders. The café’s requirements are dynamics, with potential changes such as pricing, adding new menu items, or additional features.

**CHAPTER IV**

**DEVELOPMENT, TESTING AND EVALUATION RESULT**

**Presentation of the System Output**

**Testing Results**

**CHAPTER V**

**CONCLUSION AND RECOMMENDATION**

**Conclusion**

**Recommendation**

**REFERENCES**

Kumari, M., Guleria, S., & Kumar, S. (2024). Sustainability in tourism and hospitality: Artificial intelligence role in eco-friendly practices in Indian hotels. *Journal of Tourism Theory and Research*, *10*(2), 73–83. https://doi.org/10.24288/jttr.1523976

Naidu, J. A., Kumar, S. R., Noramirah, U., Shambudin, M., & Masandig, H. (2024). *Dynamic Bookstore Self-Ordering Kiosk System*. *5*(1), 247–255.

Susanti, A., Diani, R., Satiyarti, R. B., Asyhari, A., & Destiana, A. (2024). A bibliometric analysis of MOOCs research using VOSViewer. *AIP Conference Proceedings*, *3058*(1), 0–4. https://doi.org/10.1063/5.0201266

Wei, Y., & Simay, A. E. (2023). AI Adoption in the Chinese Food and Beverage Industry: An Exploratory Study. *FIRM Journal of Management Studies*, *8*(2), 145. https://doi.org/10.33021/firm.v8i2.4412

Yee, L. J., Chan, S. W., Ismail, F., Ahmad, M. F., Ruslan, R., Ramlan, R., Nojumuddin, N. S., & Zaman, I. (2024). Factor Affecting Customers’ Acceptance of Self-service Restaurant Ordering System (SROS) in Johor Restaurant. *Research in Management of Technology and Business*, *5*(1), 957–974.

**APPENDICES**