ASSESSMENT COVERSHEET

Attach this coversheet as the cover of your submission. All sections must be completed.

Section A: Submission Details

Programme	:	Bachelor of Information Technology (Hons.) in Software Engineering
Course Code & Name	:	ISB37804 - Reuse and Component Based Development
Course Lecturer(s)	:	Dr. Azizah Binti Rahmat
Submission Title	:	Enterprise Application: Food Ordering System
Deadline	:	Day 17 Month 1 Year 2025 Time
Penalties	:	• 5% will be deducted per day to a maximum of four (4) working days, after which the submission will **not** be accepted. • Plagiarised work is an Academic Offence in University Rules & Regulations and will be penalised accordingly.

Section B: Academic Integrity

Tick (√) each box below if you agree:

√	I have read and understood the UniKL's policy on Plagiarism in University Rules & Regulations.
√	This submission is my own, unless indicated with proper referencing.
√	This submission has not been previously submitted or published.
√	This submission follows the requirements stated in the course.

Section C: Submission Receipt

(must be filled in manually)

Office Receipt of Submission

Date & Time of Submission (stamp)	Student Name(s)	Student ID(s)
17 January 2025	SAFIRAH BINTI TAJUL ARIFFIN AMIRUL EHSAN BIN ROSLAN NUR AFEEQAH INSYEERAH BINTI RAZALI NADIATUL ALYA BINTI JASMIR	52213223186 52213223261 52213122136 52213223068

Student Receipt of Submission

This is your submission receipt, the only accepted evidence that you have submitted your work. After this is stamped by the appointed staff & filled in, cut along the dotted lines above & retain this for your record.

Date & Time of Submission (stamp)	Course Code	Submission Title	Student ID(s) & Signature(s)
17 January 2025	ISB37804	Enterprise Application: Food Ordering System	52213223186 52213223261 52213122136 52213223068

Contents

1.0 Introduction

This document outlines the framework and functionalities of an advanced web-based application designed for restaurant order management. The application focuses on facilitating efficient communication between customers and restaurant operations, which includes ordering, kitchen processing, and administrative management.

1.1 Background of the System

In an increasingly digital world, the restaurant industry is adopting technology to enhance operational efficiency and customer engagement. A dedicated digital ordering system offers significant benefits by allowing customers to view menus, place orders, and make payments from their devices, minimizing wait times and improving service accuracy. The system discussed in this document leverages a combination of modern web technologies to ensure a seamless operation from the customer's order placement to the final delivery.

System components include:

Client-Side Development: Utilizes JavaServer Pages (JSP) or JavaServer Faces (JSF) for constructing the user interface, complemented by CSS for aesthetic design and JavaScript for interactive elements like dynamic menu updates and cart management.

- **Backend Processing:** Features servlets that facilitate the interaction between the frontend interface and the database, managing data flow securely and efficiently.

- **Database Operations:** Relies on a relational database system to organize and maintain critical data including user details, order records, and inventory statuses.

- **Application Logic:** Deployed using Enterprise JavaBeans (EJB), this layer handles the core business processes, ensuring data integrity and transaction security.

- **Reusable Components:** The application architecture includes shared libraries that support code reusability and system modularity, simplifying future upgrades and maintenance.

1.2 Problem Statement

The integration of effective digital solutions in the restaurant sector remains a challenge despite the availability of technology. Issues commonly faced by establishments include:

- **Efficiency in Order Handling:** Inaccuracies in order handling can negatively impact customer satisfaction and operational cost-efficiency.

- **Management of Menus and Stock:** Efficiently managing menu options and stock levels requires robust systems that can update in real-time.

- **Enhancing Customer Interactions:** Creating a smooth and engaging customer interface is vital for securing repeat business and enhancing user satisfaction.

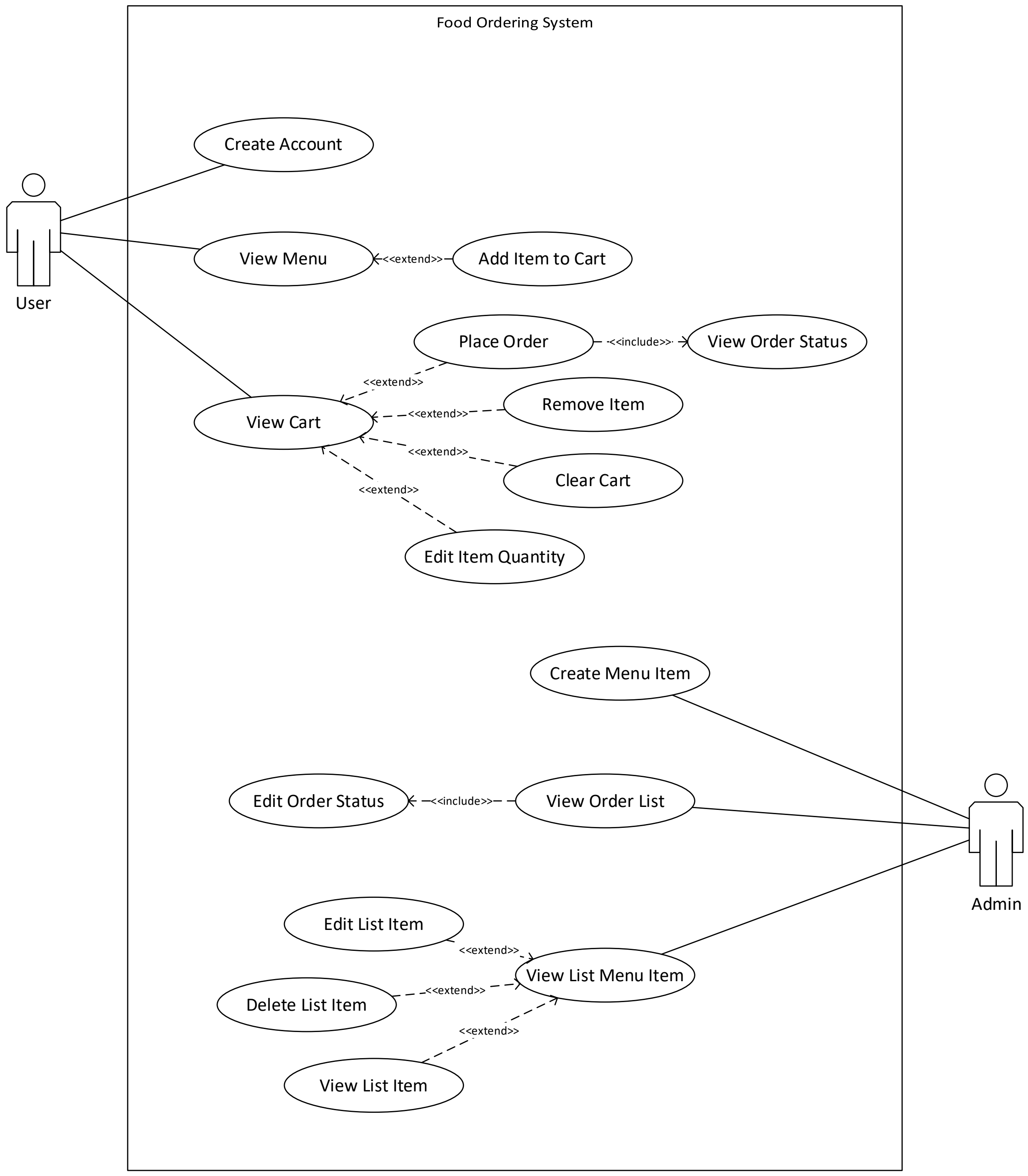
- **Security and Versatility in Payments:** Providing secure and versatile payment options is critical to accommodate diverse customer preferences.

The restaurant ordering system designed here aims to mitigate these challenges by introducing an efficient, user-friendly interface that manages orders accurately, maintains up-to-date inventory and menu information, and provides secure and flexible payment methods, ultimately improving the efficiency of restaurant operations and customer satisfaction.

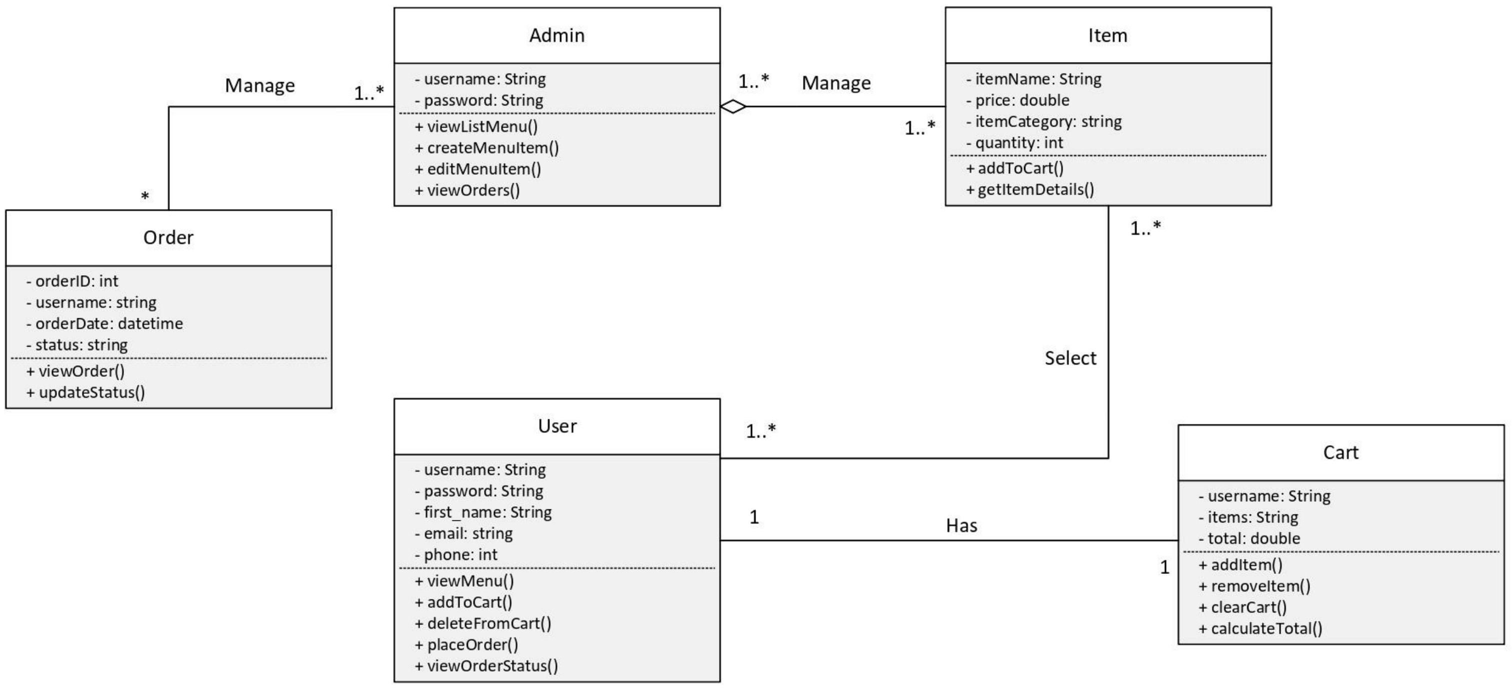
2.0 Designing the Application

2.1 Gathering user Requirements

2.1.1 Use Case Diagram

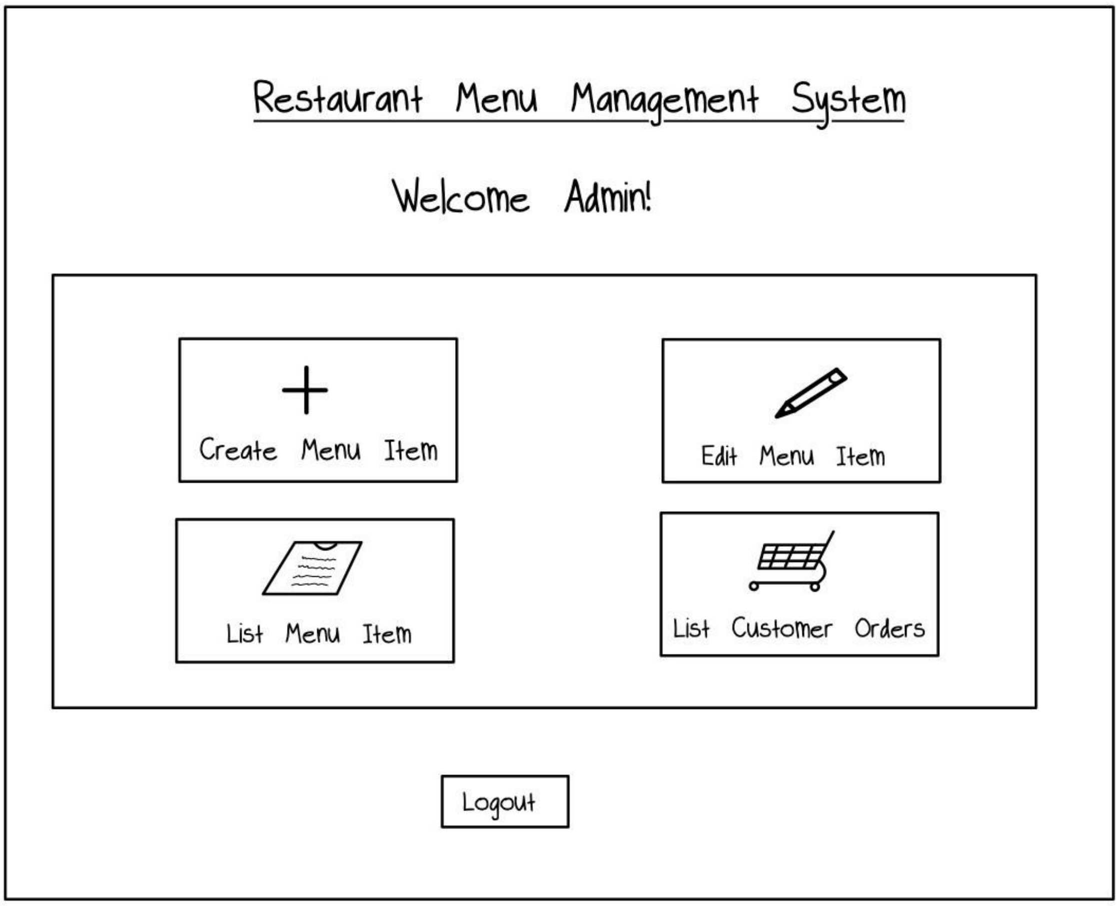


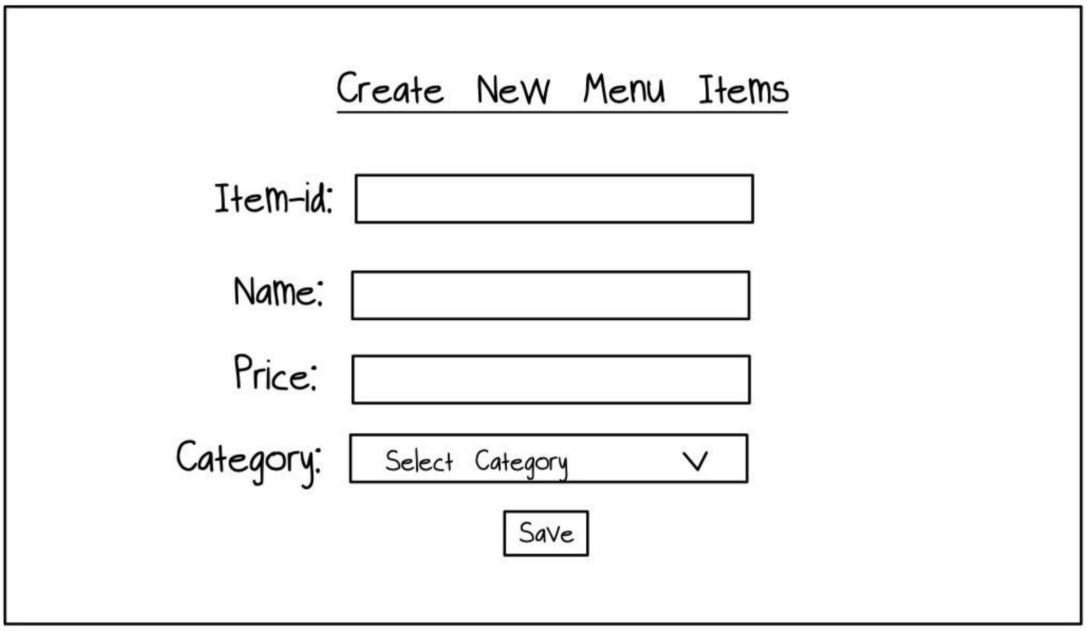
2.1.2 Class Diagram



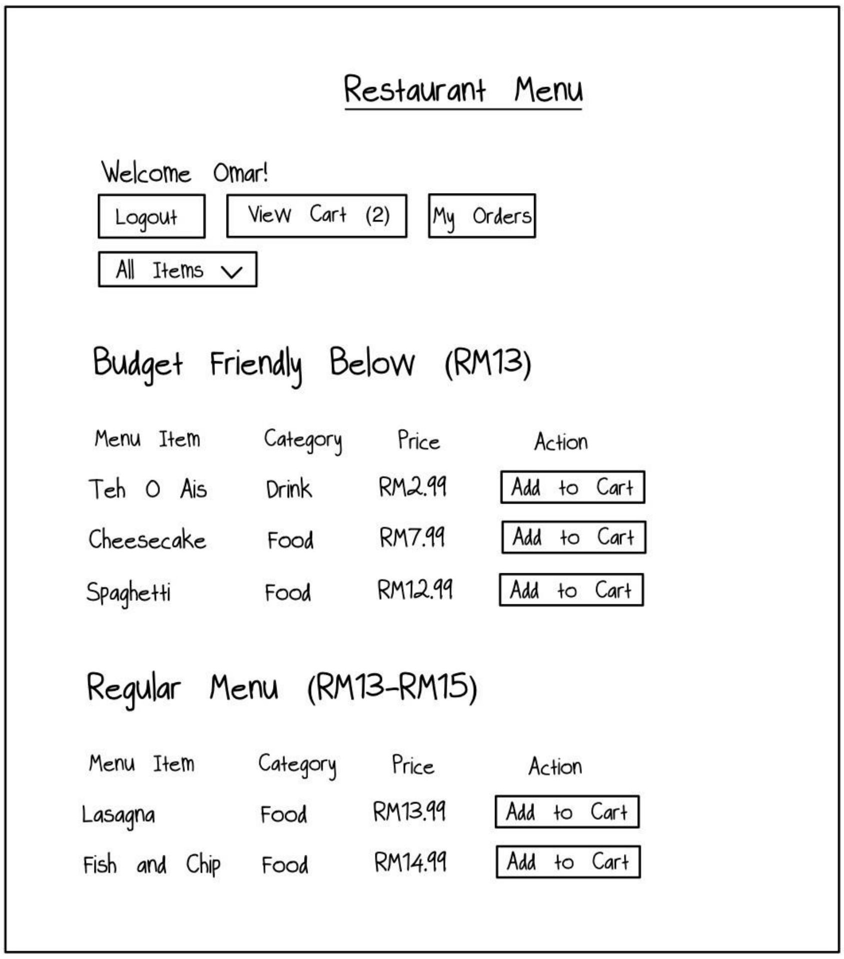
2.2 Preparing mockup

2.2.1 Admin User Interface Mockup





2.2.2 Customer User Interface Mockup

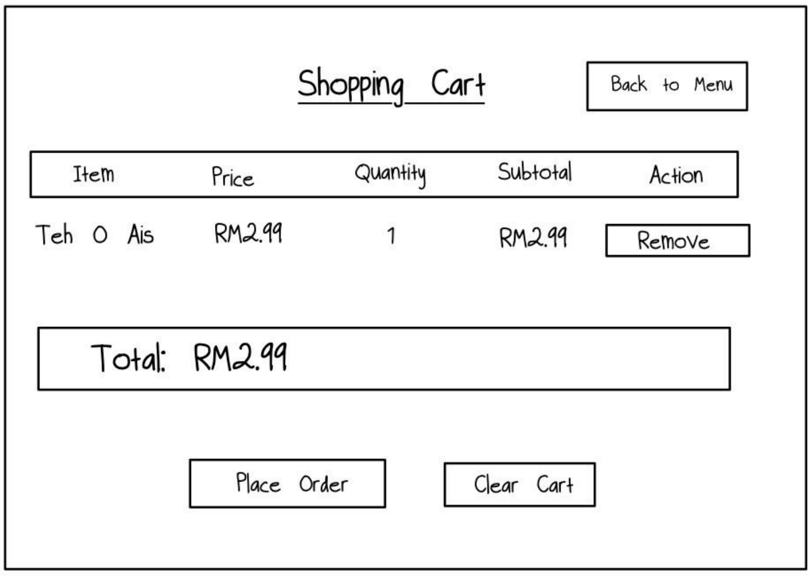


Budget Friendly Below (RM13)

Menu Item	Category	Price	Action
Teh O Ais	Drink	RM2.99	Add to Cart
Cheesecake	Food	RM7.99	Add to Cart
Spaghetti	Food	RM12.99	Add to Cart

Regular Menu (RM13-RM15)

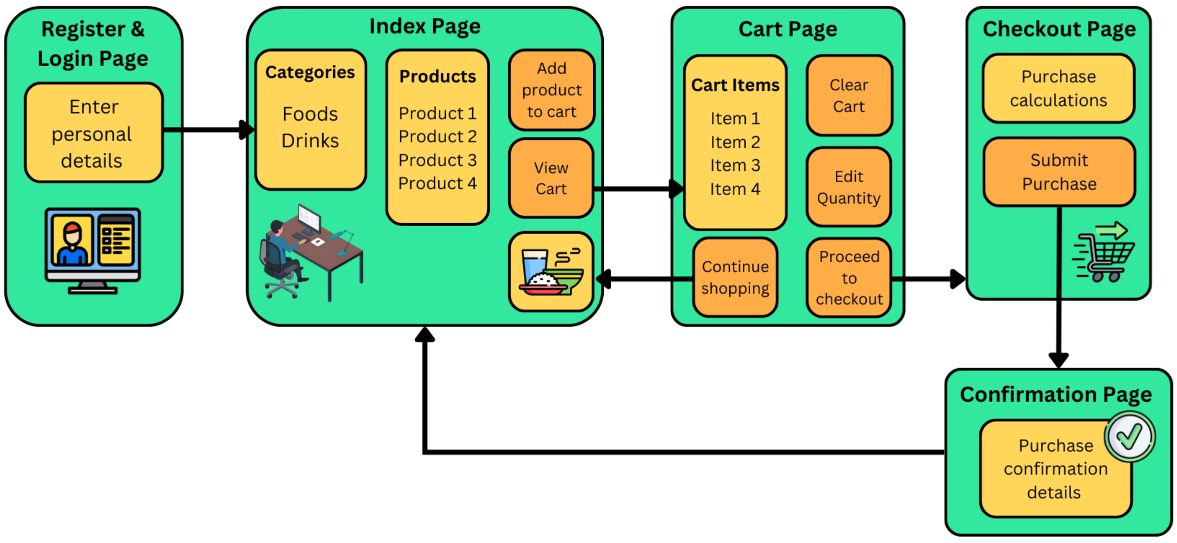
Menu Item	Category	Price	Action
Lasagna	Food	RM13.99	Add to Cart
Fish and Chip	Food	RM14.99	Add to Cart



Item	Price	Quantity	Subtotal	Action
Teh O Ais	RM2.99	1	RM2.99	Remove

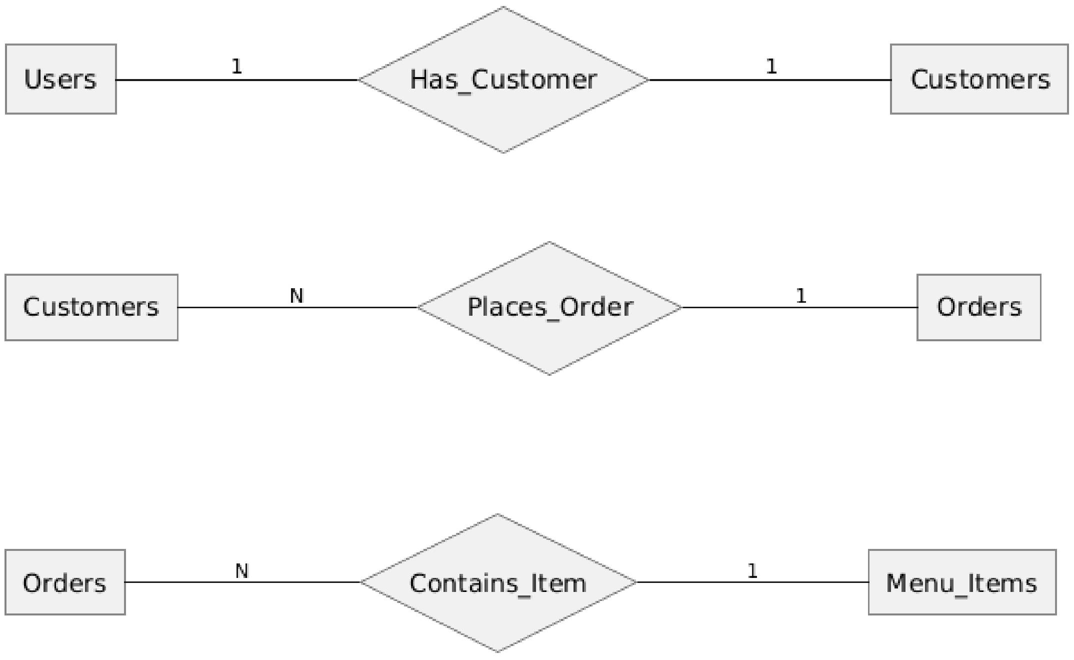
Total: RM2.99

2.3 The Business Process Flow

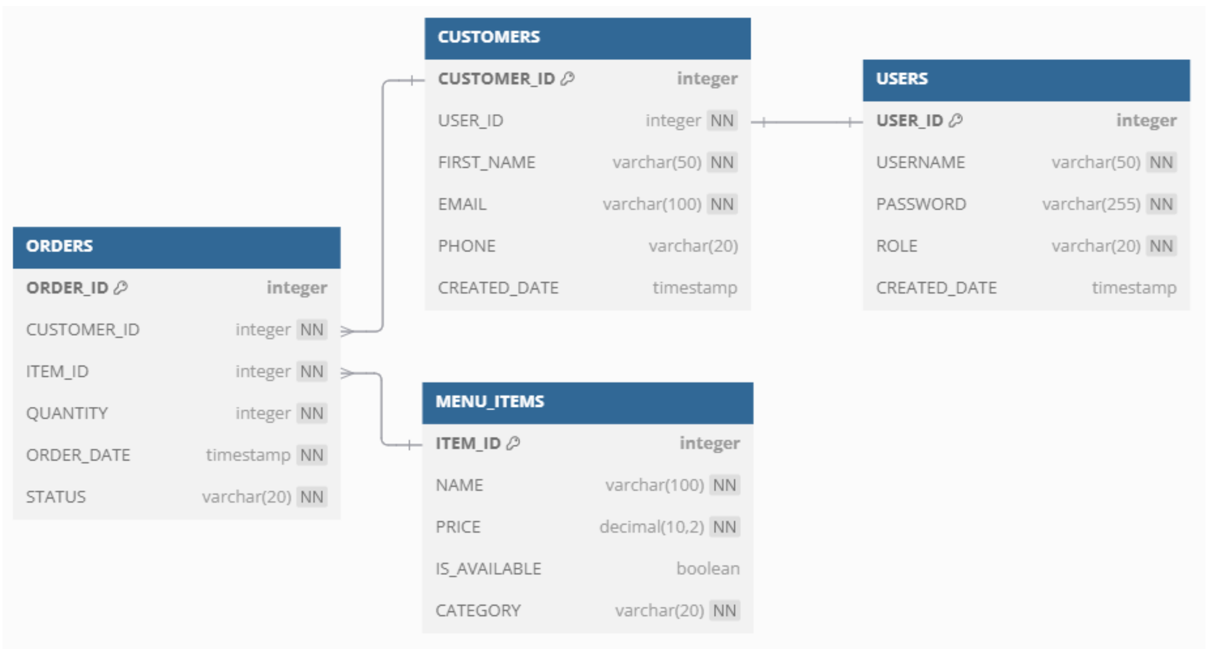


2.4 Designing the Data Model

- ER Diagram



- Data model



CUSTOMERS		
CUSTOMER_ID	integer	
USER_ID	integer	NN
FIRST_NAME	varchar(50)	NN
EMAIL	varchar(100)	NN
PHONE	varchar(20)	
CREATED_DATE	timestamp	

USERS		
USER_ID	integer	
USERNAME	varchar(50)	NN
PASSWORD	varchar(255)	NN
ROLE	varchar(20)	NN
CREATED_DATE	timestamp	

ORDERS		
ORDER_ID	integer	
CUSTOMER_ID	integer	NN
ITEM_ID	integer	NN
QUANTITY	integer	NN
ORDER_DATE	timestamp	NN
STATUS	varchar(20)	NN

MENU_ITEMS		
ITEM_ID	integer	
NAME	varchar(100)	NN
PRICE	decimal(10,2)	NN
IS_AVAILABLE	boolean	
CATEGORY	varchar(20)	NN

3.0 Future Recommendations/ Suggestions for the Food Ordering System

1. **Simplified Navigation**
   Redesign the menu and ordering interface to enhance user experience

   Develop a more intuitive and user-friendly interface to ensure that navigating the menu and placing orders is straightforward for all users. This may involve reorganizing the layout, simplifying the steps involved in placing an order, and enhancing visual cues that guide the user throughout the ordering process.

2. **Exclusive Discounts and Coupons**
   Boost customer engagement and increase order frequency

   Implement a feature that offers exclusive discounts and coupons to customers, such as limited-time deals or special offers for loyal customers. Providing these exclusive benefits creates a sense of exclusivity and incentivizes customers to place orders more frequently, ultimately improving customer retention and sales.

3. **Feedback Mechanism**
   Improve service quality and customer satisfaction

   Implement a feedback feature that allows customers to rate their orders and provide comments directly through the platform. This feedback will be invaluable for monitoring and enhancing the quality of service and food, as well as for making adjustments based on customer preferences and concerns.

4. **View Menu Ingredients**
   Support informed food choices

   Add a feature that displays detailed ingredients for each menu item. This enhancement is crucial for customers with allergies, dietary restrictions, or specific food preferences, enabling them to make safe and informed decisions when selecting meals.

5. **Implement a Payment System**
   Streamline the ordering process.

Develop and integrate a secure payment gateway that supports various payment methods, including credit cards, e-wallets, and online banking. This system should provide a seamless and secure transaction process, from order placement to payment, enhancing the overall customer experience and operational efficiency.

4.0 Conclusion

This project offered much depth about reuse and component-based development. From this project, we understood the concept of modularity in terms of building systems that can support the expanding needs. This way, it was easier to avoid repetition, and we were also lowering the stakes for future modifications. The project emphasized the necessity of the structured development. Application of well understood components made the system more maintainable. It also made the debugging and testing phase easier. This approach was less time consuming and cost effective while at the same time delivering good quality outcomes. From here, we received practical experience in assembly of prebuilt component pieces. This process involved compatibility and making adjustment of the compatibility to suit the needs of the system. It provided importance to the need to standardize the software. We also looked at the issue of success factors balancing reuse and was due for customization as well. Each component could not be reused without modification in many cases. These were changed and lengthened to cover other needs as well. It also enabled us to develop our problem-solving capabilities. This project was very much collaborative in nature. To make things more understandable, it has been discovered that the sharing of components and ideas within the team also made the process easier for development. It highlighted that good communication is key as well as good cooperation with other members of the group. That is why this project showed how reuse can be used in practice allowing applications daily. Did so affect the quality and was cumulatively beneficial by cutting development costs and improving efficiency. In general, it enhanced the comprehension of component-based development to a great extent. It helped us develop skills in constructing systems that are generic and reusable, easily scalable and maximize resource usage. This knowledge will be our guide for future software engineering projects.

5.0 Appendix

1. Menu.xhtml - page to see the menu page

```
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:f="http://xmlns.jcp.org/jsf/core">
    <h:head>
        <title>Restaurant Menu</title>
        <link rel="stylesheet" type="text/css" href="resources/css/style.css"/>
    </h:head>
    <h:body>
        <div class="main-container">
            <div class="header">
                <h1>Restaurant Menu</h1>
                <div class="user-info">
                    <span>Welcome, #{loginController.currentUser.username}!</span>
                    <h:form>
                        <h:commandButton value="Logout"
                                       action="#{loginController.logout}"
                                       styleClass="btn-secondary"/>
                        <!-- Add View Cart Button -->
                        <h:commandButton value="View Cart
(#{cartController.cartItems.size()})"
                                       action="cart.xhtml"
                                       styleClass="btn-primary"/>
                        <h:commandButton value="My Orders"
                            action="myOrders.xhtml"
                            styleClass="btn-primary"/>
                    </h:form>
                </div>
            </div>

            <!-- Messages Panel -->
            <h:panelGroup id="messagePanel" layout="block">
                <h:messages errorStyle="color: red" infoStyle="color: green"
layout="table"/>
            </h:panelGroup>

            <div class="menu-container">
                <!-- Category Selector Form -->
                <h:form id="categoryForm">
                    <div class="category-selector">
                        <h:selectOneMenu value="#{customerController.selectedCategory}"
                                       onchange="submit()">
                            <f:selectItem itemValue="ALL" itemLabel="All Items"/>
                            <f:selectItem itemValue="FOOD" itemLabel="Food Only"/>
                            <f:selectItem itemValue="DRINK" itemLabel="Drinks Only"/>
                        </h:selectOneMenu>
                    </div>
                </h:form>

                <!-- Budget Friendly Section -->
                <h:form id="budgetForm">
                    <h2>Budget Friendly (Below RM13)</h2>
                    <h:dataTable value="#{customerController.budgetItems}" var="item"
                                styleClass="menu-table">
                        <h:column>
                            <f:facet name="header">
                                <h:outputText value="Menu Item"/>
                            </f:facet>
                            <div class="menu-item-name">#{item.name}</div>
                        </h:column>
                        <h:column>
                            <f:facet name="header">
                                <h:outputText value="Category"/>
                            </f:facet>
                            <div class="menu-item-category">#{item.category}</div>
                        </h:column>
                        <h:column>
                            <f:facet name="header">
                                <h:outputText value="Price"/>
                            </f:facet>
                            <div class="menu-item-price">
                                <h:outputText value="#{item.price}">
                                    <f:convertNumber type="currency" currencySymbol="RM"/>
                                </h:outputText>
                            </div>
                        </h:column>
                        <h:column>
                            <f:facet name="header">
                                <h:outputText value="Action"/>
                            </f:facet>
                            <h:commandButton value="Add to Cart"
                                        action="#{cartController.addToCart(item)}"
                                        styleClass="btn-primary">
                                <f:ajax render="messagePanel"/>
                            </h:commandButton>
                        </h:column>
                    </h:dataTable>
                </h:form>

                <!-- Regular Menu Section -->
```

```
<h:form id="regularForm">
    <h2>Regular Menu (RM13 - RM15)</h2>
    <h:dataTable value="#{customerController.regularItems}" var="item"
            styleClass="menu-table">
        <h:column>
            <f:facet name="header">
                <h:outputText value="Menu Item"/>
            </f:facet>
            <div class="menu-item-name">#{item.name}</div>
        </h:column>
        <h:column>
            <f:facet name="header">
                <h:outputText value="Category"/>
            </f:facet>
            <div class="menu-item-category">#{item.category}</div>
        </h:column>
        <h:column>
            <f:facet name="header">
                <h:outputText value="Price"/>
            </f:facet>
            <div class="menu-item-price">
                <h:outputText value="#{item.price}">
                    <f:convertNumber type="currency" currencySymbol="RM"/>
                </h:outputText>
            </div>
        </h:column>
        <h:column>
            <f:facet name="header">
                <h:outputText value="Action"/>
            </f:facet>
            <h:commandButton value="Add to Cart"
                    action="#{cartController.addToCart(item)}"
                    styleClass="btn-primary">
                <f:ajax render="messagePanel"/>
            </h:commandButton>
        </h:column>
    </h:dataTable>
</h:form>

<!-- Premium Menu Section -->
<h:form id="premiumForm">
    <h2>Premium Menu (Above RM15)</h2>
    <h:dataTable value="#{customerController.premiumItems}" var="item"
            styleClass="menu-table">
        <h:column>
            <f:facet name="header">
                <h:outputText value="Menu Item"/>
            </f:facet>
```

```
                    <div class="menu-item-name">#{item.name}</div>
                </h:column>
                <h:column>
                    <f:facet name="header">
                        <h:outputText value="Category"/>
                    </f:facet>
                    <div class="menu-item-category">#{item.category}</div>
                </h:column>
                <h:column>
                    <f:facet name="header">
                        <h:outputText value="Price"/>
                    </f:facet>
                    <div class="menu-item-price">
                        <h:outputText value="#{item.price}">
                            <f:convertNumber type="currency" currencySymbol="RM"/>
                        </h:outputText>
                    </div>
                </h:column>
                <h:column>
                    <f:facet name="header">
                        <h:outputText value="Action"/>
                    </f:facet>
                    <h:commandButton value="Add to Cart"
                               action="#{cartController.addToCart(item)}"
                               styleClass="btn-primary">
                        <f:ajax render="messagePanel"/>
                    </h:commandButton>
                </h:column>
            </h:dataTable>
        </h:form>
    </div>
</div>
</h:body>
</html>
```

2. Myorder.xhtml page – to check the order have places

```
<?xml version="1.0" encoding="UTF-8" ?>
        <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
        <html xmlns="http://www.w3.org/1999/xhtml"
            xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
            xmlns:h="http://xmlns.jcp.org/jsf/html"
```

```
    xmlns:f="http://xmlns.jcp.org/jsf/core">
  <ui:composition template="/template.xhtml">
    <ui:define name="title">
      <h:outputText value="My Orders"/>
    </ui:define>
    <ui:define name="body">
      <div class="orders-container">
        <h:form>
          <div class="order-summary">
            <div class="status-count">
              <span class="status-pending">
                Pending: #{customerController.orderStatusCounts['PENDING']}
              </span>
              <span class="status-processing">
                Processing:
#{customerController.orderStatusCounts['PROCESSING']}
              </span>
              <span class="status-completed">
                Completed:
#{customerController.orderStatusCounts['COMPLETED']}
              </span>
              <span class="status-cancelled">
                Cancelled:
#{customerController.orderStatusCounts['CANCELLED']}
              </span>
            </div>
          </div>

          <h:panelGroup id="ordersPanel">
            <h:messages globalOnly="true" styleClass="messages"
                   errorClass="error-message"
                   infoClass="info-message"/>

            <h:outputText value="No orders found." rendered="#{empty
customerController.customerOrders}"
                    styleClass="no-orders-message"/>

            <h:dataTable value="#{customerController.customerOrders}"
                    var="order"
                    styleClass="orders-table"
                    headerClass="orders-header"
                    rowClasses="order-row-odd,order-row-even"
                    columnClasses="col-id,col-item,col-quantity,col-date,col-
status,col-actions"
                    rendered="#{not empty customerController.customerOrders}">

              <h:column>
                <f:facet name="header">Order ID</f:facet>
```

```
                    <h:outputText value="#{order.orderId}"/>
                </h:column>

                <h:column>
                    <f:facet name="header">Item</f:facet>
                    <h:outputText
value="#{customerController.getItemName(order.itemId)}"/>
                </h:column>

                <h:column>
                    <f:facet name="header">Quantity</f:facet>
                    <h:outputText value="#{order.quantity}"/>
                </h:column>

                <h:column>
                    <f:facet name="header">Order Date</f:facet>
                    <h:outputText
value="#{customerController.getFormattedDate(order.orderDate)}"/>
                </h:column>

                <h:column>
                    <f:facet name="header">Status</f:facet>
                    <h:outputText value="#{order.status}"

styleClass="#{customerController.getStatusStyle(order.status)}"/>
                </h:column>

                <h:column>
                    <f:facet name="header">Actions</f:facet>
                    <h:commandButton value="Cancel Order"
                                action="#{customerController.cancelOrder(order)}"
                                styleClass="btn-danger"

rendered="#{customerController.canCancelOrder(order)}"
                                onclick="return confirm('Are you sure you want to
cancel this order?');">
                        <f:ajax render="ordersPanel"/>
                    </h:commandButton>
                </h:column>
            </h:dataTable>
        </h:panelGroup>

        <div class="actions-container">
            <h:commandButton value="Back to Menu" action="menu"
styleClass="btn-secondary"/>
            <h:commandButton value="Refresh Orders"
                        action="#{customerController.refreshOrders}"
                        styleClass="btn-primary">
                <f:ajax render="ordersPanel"/>
              </h:commandButton>
            </div>
          </h:form>
        </div>
      </ui:define>
    </ui:composition>
</html>
```

6.0 References

- Admin, Admin, Admin, & Admin. (2024b, February 13). Navigating the Challenges of Online food Ordering: Tips for success. Online eMenu : Blog. https://www.onlineemenu.com/blog/navigating-the-challenges-of-online-food-ordering-tips-for-success/

- Padkowska, A. K.-. (2025b, January 8). How to set up online ordering system for your restaurant (10 steps). UpMenu. https://www.upmenu.com/blog/how-to-set-up-online-ordering-system/

- Ecommerce UX: design Strategies and best Practices for 2024. (2023b, November 27). Shopify. https://www.shopify.com/my/blog/ecommerce-ux

- Ashraf, A. (2024, January 22). Boost your online food ordering system: 10 powerful promotion tactics. WPExperts. https://wpexperts.io/blog/boost-online-ordering-system/