1.0 COURSEWORK TITLE

University Food Ordering System

2.0 THE COURSEWORK OVERVIEW

Tech university would like to develop an application that streamline the food order taking process by connecting different types of users such as cafeteria vendor, customer, and administrator. By using the application, customers can order food based on the menus provided by various vendors from the cafeteria. Customers can choose to dine in, take-away or request for delivery service through the application. Once an order is created, the application will promptly alert the vendor to proceed with food preparation and inform the customers when the order ready. Furthermore, the application has its own credit-based payment system which allows customer to pay for the order. Customer can reload the credit via administrator.

The application should have the following features:

- Login access
- User Registration
- Menu
- Food order placement
- Notification
- Payment
- Order History
- Delivery System

In addition, a supporting document is needed to reflect the design of the implementation codes and the implementation details that utilize the Object-oriented programming concepts.

3.0 OBJECTIVES OF THIS COURSEWORK

Develop the practical ability to describe, justify, and implement an Object-oriented system.

4.0 TYPE

Group Assignment (minimum 3 and maximum 4 students)

5.0 COURSEWORK DESCRIPTION

As an object-oriented programming student, you are required to identify the relationship among the entities and develop a user-friendly application for different types of users by fulfilling the basic features that listed in section 2.0.

5.1 Login access:

The application should be designed with four types of access rights such as vendor, customer, delivery runner, and administrator. Table 1.0 shows the functionalities that can be accessed by each user upon login.

Roles	Functionalities
Vendor	❖ Create/Read/Update/Delete item ❖ Accept/Cancel order ❖ Update order status ❖ Check order history **daily, monthly, quarterly, etc.* ❖ Read customer review ❖ Revenue Dashboard
Customer	❖ View menu ❖ Read customer review ❖ Place/Cancel order ❖ Check order status ❖ Check order history ❖ Check transaction history ❖ Provide a review for each order ❖ Reorder using order history
Delivery runner	❖ View task ❖ Accept/Decline task ❖ Update task status ❖ Check task history ❖ Read customer review ❖ Revenue Dashboard
Administrator	❖ User registration ▪ Create/Read/Update/Delete vendor ▪ Create/Read/Update/Delete customer ▪ Create/Read/Update/Delete runner ❖ Top-up customer credit ❖ Generate transaction receipt ❖ Send receipt to customer through notification

Table 1.0

5.2 User Registration

Your program should only allow administrators to register customers and vendors before they can use the application. User details need to be saved in a text file.

5.3 Menu

As there are multiple cuisines offered at Tech university, the application should present menus from all vendors to make it easier for customers to place order using the application.

5.4 Food order placement

One of the main objectives of this application is to make the food ordering process easier. Your program should allow customers to order food through the application by browsing the menu offer by various vendors. Customers can choose to dine-in, takeaway or request for delivery when placing the order. Additional charges will be imposed if customers request for delivery service. Besides, the application streamlines day-to-day operations of vendors by receiving and managing orders. Vendors can choose to either accept or decline the order and if an order is declined, the application will refund to the customers.

5.5 Payment

The application should incorporate a digital wallet feature. The purpose of it is to allow customers to pay when they place an order. This feature will automatically deduct the required amount from customer's credit for each order. The application should prevent customers from placing the order if the credit is insufficient for it. Customers can reload their credit via administrators. Hence, your program should include a feature that enable administrators to update customers' credit based on their account ID, followed by sending a digital receipt to customer via notification system.

5.6 Notification

The in-app notification system is designed for all types of users. As mentioned in Section 5.5, administrators can use it to send digital receipt every time customers reload their account credit. Besides, it is also important to keep both vendors and customers updated about their order. For example, when a customer places an order, the vendor will receive a notification and the application will notify customer whether the vendor has accepted or declined the order.

5.7 Order History

This feature should be enabled for both customers and vendors as it provides the overview of all orders and transactions. Customers can track the status and details of their current and past orders such as order placement times, order amount, acceptance or declined status, and so on. Customers should have the option to reorder using their order history. Meanwhile, vendors can keep track of their revenue and produce report based on the order history. Furthermore, vendors can also review customer feedback and ratings for every order.

5.8 Delivery System

This feature is designed to simulate the delivery process. As mentioned earlier, customers are allowed to choose delivery service when placing an order. The application will allocate a runner automatically and send them notification to keep them notified with the order status. The runner can choose to accept or decline the task and when the task is declined, the application must allocate next available runner. If there is no available runner, the application must notify customers and prompt them to choose either dine-in or takeaway. The delivery fees will be credit back to the customer's account. On the other hand, there are some basic features that need to be developed in the application for runners such as view task, update task statuses, view task history, and so on. Furthermore, runners should be able to track their daily, monthly, or yearly earnings using the application.

6.0 General Requirements

- The program submitted should compile and be executed without errors
- User input validation should be done to avoid logical error

- The implementation code must highlight the use of Object-oriented programming concepts
- Students should use text files for storing and retrieving data required for the system
- Database tools like access, oracle, sql servers, etc. are not allowed

7.0 Deliverables

- Report and source code
- Submission deadline: 8th December 2023, 11:59pm

7.1 Report:

The report needs to be submitted in softcopy to Moodle with:

A. Cover Page
   - Module
   - Coursework Title
   - Intake code
   - Student name and id
   - Date Assigned (The date the report was handed out)
   - Date Completed (The date the report is due to be handed in)

B. Table of Contents

C. Contents
   - Design solution (Use Case Diagram and Class Diagram)
   - Screenshots of output of the program with appropriate explanations
   - Description and justification of Object-oriented concepts incorporated into the solution
   - Additional feature

D. Limitation and Conclusion

E. References

F. Appendix
   - Screenshots of all text files that are used to store the data

7.2 Source Code:

Application source code need to be compiled into zip folder and submitted to Moodle.

8.0 Assignment Assessment Criteria

The assignment assessment consists of two main components: System Implementation (40%) and System Documentation (15%).

Assessment	Description	Weightage (%)
System Implementation	Performance result of all operations and design per requirements	16
	Appropriate design and the implementation of codes illustrating the object-oriented programming concept incorporated	16

	Presentation	8
System Documentation	Description and justification of the object-oriented concepts incorporate	6
	UML Diagrams	4
	Program output screenshots	3
	Report format and references	2

9.0 Development Tools

The program must be written in Java language, and you can use any Java development IDE as a tool, but back-end data must be stored in **.txt** files.

10.0 Academic Integrity

You are expected to maintain the utmost level of academic integrity during the duration of the course. Plagiarism is a serious offence and will be dealt with according to regulations of Asia Pacific University on plagiarism.