Report Structure:

1. Introduction

- **Brief overview of the project: RestaurantOrderingSystem**

- **Motivation behind choosing this project:**

  Making a Restaurant Ordering system was so fun, and learn a lot about class functions and void calls. This is the one part of improving C++ coding

- **Technologies/libraries used:**
  - #include <iostream>
  - #include <iomanip>
  - #include <string>
  - #include <vector>
  - #include <cstdlib>
  - using namespace std

2. Project Objectives

- **List the main goals of the project.**
  - Welcome Customer Admin
  - Display Food Menu Drink Menu
  - Take Order
  - Display Current Order
  - Add more items
  - Cancel an item
  - Checkout
  - Final Bill Summary
  - Manage Menu
  - Add Item into menu
  - Remove Item from menu
  - Each section print new output clear old output after input
  - GUI for system

- **Describe what the system/game should accomplish.**
  - ➢ Welcome Customer Admin
    1. Get an Order (Customer)
       - Display Food & Drink Menu
         - Enter Food/Drink ID (Input ID number in the Menu if wrong, will ask for Input again)
           - Enter quantity (Input only number of quantity if wrong, will ask for Input again (start from Enter Food/Drink ID)
             - Quantity(number) x Item(name) added.
             - Subtotal: $ (TotalBill)
         - 0 to Back
         - 168 To Continue
           - Display Current Order
             1. Add more items (Go back to Display Food & Drink Menu and Enter Food/Drink ID)
             2. Cancel an item
                - If no items in the order to cancel (Go back to Display Current Order and ask again, Add more items or Cancel an item or Checkout)
                - Enter O.N(order number) of item to remove(Input O.N(order number) only If wrong, will ask for Input again, go back to Display Current Order and ask again, Add more items or Cancel an item or Checkout)
                  - Food item added successfully!
                - 0 to cancel (Go back to Display Current Order and ask again, Add more items or Cancel an item or Checkout)
             3. Checkout
                - Final Bill Summary (End)

    2. Manage Menu (Admin)
       - Display Food Menu Drink Menu
         1. Add Item to Menu
            - Display Food & Drink Menu
              - Enter item ID (Input new ID number if wrong or ID exists, will ask for Input again)
                - Enter item name
                  - Enter item price (Input only number of price if wrong, will ask for Input again (start from Enter item ID)
                    - Is this a Food or Drink? (Input only one; food or drink. If wrong, will go back to Display Food & Drink Menu and ask Add Item to Menu or Remove Item from Menu or Back to Main Menu)
                      - Drink item added successfully
         2. Remove Item from Menu
            - If no items in the order to cancel (Go back to Display Food & Drink Menu and ask Add Item to Menu or Remove Item from Menu or Back to Main Menu)
            - Display Food & Drink Menu

        - Enter the ID of the item to remove (Input only ID number of Item from Menu if wrong or word instant, will go back to Display Food & Drink Menu and ask Add Item to Menu or Remove Item from Menu or Back to Main Menu)
    3. Back to Main Menu (Go back to Welcome Customer Admin)

3. Features & Functionalities

- **Core Features:**
  - **class MenuItem**
  - **class OrderItem**
  - **class RestaurantOrderingSystem**

    Private:
    - vector<MenuItem> FoodMenu;
    - vector<MenuItem> DrinkMenu;
    - vector<OrderItem> Orders;
    - float totalBill = 0.0;

    Public:
    - RestaurantOrderingSystem()
      FoodMenu.push_back(MenuItem(id, name, price));
      DrinkMenu.push_back(MenuItem(id, name, price));
    - void AddItem()
    - void RemoveItem()
    - void ManageMenu()
    - void WelcomeCustomerAdmin()
    - void DisplayMenu()
    - void TakeOrder()
    - void ACC()
    - void DisplayCurrentOrder()
    - void CancelItem()
    - void PrintBill()

- **Additional Features**
  - MenuItem* FindItemById(int id)
  - bool IsIdExist(int id)
  - cin.fail()
  - cin.clear()
  - cin.ignore(10000000000)

- system("cls") system("pause")
- FoodMenu.empty() DrinkMenu.empty()
- FoodMenu.begin() FoodMenu.end()
- DrinkMenu.begin() DrinkMenu.end()

4. Challenges and Limitations

1. Group discussions at first are so active and then go silent day by day only seen and one or two people responding.
   - Chat privately with each person.

2. At first we only wanted only one system for customers but later we decided make one more for admin system, it's hard for us to do code group and teacher since last semester till now didn't teach how to use git and most of members them don't know how to use and the code write style hard to understand and don't give specific name of int or string.
   - One or two people try to understand GitHub and some people give an idea and run tests on the code. Two people do the coding, one person combines the idea and code and makes it. When we have bugs, we fix bugs together.

3. Each output looks so messy and confusing; it does not look as good as what we expect.
   - We search on Google and YouTube on how to make it clear every time after input and printing the new output.

4. Do flowcharts seem a little bit hard and complicated.
   - We try to understand the flow and make it.

5. GUI
   For C++, there were two frameworks we found that could be used to make the GUI. One is wxWidgets. While the other is qt. both of them have their pros and cons, but we did not use any of them, because of time constraints and limited knowledge of the framework.
   - So we decided to just use the terminal instead of the program.

5. Member Roles and Responsibilities

1. **VITHANITH VORN**
    1. Mind Idea of the project
    2. Make a system for customers
    3. Combining code between customers and admin
    4. Fix the code
    5. Fix and check the Flowchart
    6. Write Report Structure
    7. Support behind the presentation
    8. Demo test the program when presentation

2. **Daravichhey Sokha**
    1. Idea of the project
    2. Make a system for admin
    3. Test the code
    4. Check the Flowchart
    5. Discuss Report Structure
    6. Support behind the presentation
    7. Demo test the program when presentation

3. **Sothyrak Duong**
    1. Test the code
    2. Make Flowchart
    3. Presentation code

4. **Sharif Shefiy**
    1. Test the code
    2. Find bugs
    3. Make Flowchart
    4. Presentation code

5. **Phanith Him**
    1. Test the code
    2. Make Flowchart
    3. Presentation code