Final Project: Point of Sale System for a Restaurant

Project Overview

Design and implement a Point of Sale (POS) system for a restaurant. This system will allow the restaurant to **manage orders**, **calculate costs and revenue**, and **generate reports on profitability**. The focus of this project is to use modern C++ techniques and demonstrate good design practices while keeping the implementation modular.

Requirements

1. **Data Storage:**
   - Use STL vectors instead of arrays or dynamic arrays for storing menu items, orders, and other relevant data.
2. **Menu Items:**
   - The menu includes:
     - Hamburger
     - Cheeseburger
     - Fries
     - Soda
     - Milkshake
   - Each menu item should have:
     - A **name** (e.g., "Hamburger," "Cheeseburger").
     - A **cost** (the amount it costs the restaurant to prepare the item).
     - A **sales price** (the price charged to the customer).
3. **Accepting Orders:**
   - The system should allow customers to place orders by specifying multiple items and their respective quantities.
4. **Reporting:**
   - The system should generate the following reports:
     - **Total cost** of all items sold (sum of cost × quantity for each item).
     - **Total revenue** (sales price × quantity for all items).
     - Profit for each item, calculated as (sales price - cost) × quantity.
     - **Profit for the entire store**, calculated as the sum of all individual item profits.
5. **Operator Overloading:**
   - Overload operators where appropriate to simplify functionality (e.g., adding orders, printing reports, or manipulating data).

Design Guidelines

- **Modular Structure:**
  - Avoid overdesigning but maintain a clear and logical separation of concerns.
  - Use multiple `.h` and `.cpp` files to organize your code. Avoid putting all functions into `main.cpp`.
- **Demonstrate Proper Functionality:**
  - In `main.cpp`, demonstrate all functionalities, including:
    - Initializing the system with menu items and their prices/costs.
    - Adding and processing orders.
    - Generating reports for total cost, total revenue, item-wise profit, and total store profit.

Submission Requirements

1. **Write-Up:**
   - Provide a written explanation of your overall approach, design decisions, and how you implemented the features.
   - Include diagrams where appropriate (e.g., class diagrams).
2. **Code Files:**
   - Submit all `.h` and `.cpp` files for the project.
   - Your code should be modular, well-documented, and follow best practices for readability and maintainability.
3. **Output:**
   - Submit a `.txt` file with the system's output showing all required functionalities demonstrated in `main.cpp`.
   - The system should not prompt for user input but instead run a series of predefined operations to showcase its features.

Grading Criteria

1. **Design:**
   - Clarity, modularity, and organization of your design will be a significant portion of the grade.
   - Avoid overly complex designs or putting everything into a single file.
2. **Functionality:**
   - Your implementation should meet all the requirements outlined above and produce the correct results.
3. **Code Quality:**
   - Use of modern C++ techniques, such as STL vectors, appropriate operator overloading, and meaningful class structures.
4. **Write-Up:**

    - A clear and concise explanation of your approach and design choices.

Tips for Success

- Take time to think through your design before coding.
- Leverage STL vector features effectively for storing and manipulating data.
- Use meaningful names for classes, variables, and functions.
- Test your program thoroughly to ensure correctness and completeness.
- Include comments and documentation to clarify your code.