STATIC PERSPECTIVE

Decomposition style and Layer style and Uses style  
  
**1. Primary presentation:**



1. **Element catalog:**
2. Elements and their properties

|  |  |  |
| --- | --- | --- |
| Elements | | Properties |
| GUI | ProductUI | Display the product interface to the manager to control and perform some tasks such as add new product, update product, delete product. |
| CategoryUI | Display the category interface to manager to control and perform some task such as add new category, update category, delete category. Moreover, the manager can add new a product into a category or delete a product from category. |
| SynchUI | This interface which is modified synch of system by admin:   * Manual Synch: the cashier synchronizes data when needed. * Automatic Synch: the admin setting the synchronus time, and the system auto synchronized at a specified time. |
| UserAccount | This is a interface to manage user account. The admin can add new user, update or deactive user. |
| StatisticUI | This interface help the manager can make a statistics such as: a statistic about products which are bought per day/week/month/year; customers buy products on retail store frequently. |
| SalesUI | This interface help the cashier can pay a bill for customer. Select a payment method such as: pay by loyalty point mode or pay cash mode. |
| Business Logic | ProductBL | Includes functions to handle event that the user perform to do some operation related product. | |
| CategoryBL | Includes functions to handle event that the user perform to do some operation related category. | |
| StatisticBL | Includes functions to handle event that the user perform to do some operation related statistics. | |
| UserAccountBL | Includes functions to handle event that the user perform to do some operation related user account. | |
| SaleBL | Includes functions to handle event that the user perform to do some operation related sale. | |
| Common | This is the class containing some functions (or variable) are common to other classes in business logic layer. | |
| Data Transfer Object | ProductDTO | Includes variables only belong to Product (Examples: mã Product, Tên product, giá tiền, mã danh mục), không chứa các hàm khác. | |
| CategoryDTO | Chưa các biến thuộc riêng về Category (VD: mã Category, tên Category), không chứa các hàm khác. | |
| CustomerDTO | Chưa các biến thuộc riêng về Customer (VD: mã Customer, tên Customer, số điện thoại, địa chỉ, tổng điểm), không chứa các hàm khác. | |
| BillDTO | Chưa các biến thuộc riêng về việc thanh toán bill với khách hàng (VD: mã Bill, ngày thực hiện,…), không chứa các hàm khác. | |
| BillDetailDTO | Chứa các biến thể hiện thông tin chi tiết của Bill (), không chứa các hàm khác. | |
| UserAccountDTO | Chưa các biến thuộc riêng về tài khoảng người dùng (VD: UserID, password,…), không chứa các hàm khác. | |
| Data Access | UserAccountDB | Chứa các hàm xử lý việc truy xuất và lưu dữ liệu trên database liên quan đến tài khoảng người dùng | |
| ProductDB | Chứa các hàm xử lý việc truy xuất và lưu dữ liệu trên database liên quan đến sản phẩm | |
| CustomerDB | Chứa các hàm xử lý việc truy xuất và lưu dữ liệu trên database liên quan đến khách hàng | |
| CategoryDB | Chứa các hàm xử lý việc truy xuất và lưu dữ liệu trên database liên quan đến danh mục | |
| SaleDB | Chứa các hàm xử lý việc truy xuất và lưu dữ liệu trên database liên quan đến việc thanh toán hoá đơn. | |

1. Relations and their properties

|  |  |
| --- | --- |
| Connector | Properties |
| Allowed to use | Cho phép các class trong một layer này có thể sử dụng các class trong layer khác |

1. **Context diagram:**



1. **Architecture background:**

Hệ thống được chia làm 4 tier bao gồm: GUI layer, Business Logic Layer, Transfer Data Object Layer, Data Access Layer.

Khi phát triển mô hình 4 lớp sẽ giảm được thời gian bảo trì và phát triển chương trình vì chia theo mô hình đa lớp tức là giống với việc sử dụng nguyên tắc chia để trị, khi thay đổi mộ hàm nào đó ở lớp này thì nó không ảnh hưởng nhiều đến lớp khác vì vậy chi phí bảo trì thấp hơn; có thể nâng cấp hoặc sửa đổi dễ dàng hơn. Từng lớp có thể được tái sử dụng lại nhiều lần ở nhiều ứng dụng khác nhau.

Mỗi lớp có thể được chạy trên các OS khác nhau ví dụ: database có thể chạy trên UNIX, LINUX; còn GUI layer thì chạy trên Windows (hoặc Web server) và được phát triển bằng nhiều ngôn ngữ lập trình cũng như nhiều nhóm phát triển.

Giới thiệu từng lớp và mục đích sử dụng:

* GUI layer: Lớp này làm nhiệm vụ giao tiếp với người dùng cuối để thu thập dữ liệu và hiển thị kết quả/dữ liệu thông qua các thành phần trong giao diện người sử dụng. Lớp này sẽ sử dụng các dịch vụ do lớp Business Logic cung cấp. Theo phân tích yêu cầu đề bài thì lớp GUI gồm những class giao diện sau: ProductUI, CustomerUI, CategoryUI, StatisticUI, ManagementUI, SaleUI.
* Business Logic Layer: xử lý chính các dữ liệu trước khi được đưa lên hiển thị trên màn hình hoặc xử lý các dữ liệu trước khi lưu dữ liệu xuống cơ sở dữ liệu. Lớp business logic có thể kiểm tra và thực hiện có yêu cầu nghiệp vụ, các tính năng tính toán của chương trình sẽ thực thi theo một trình tự logic. Lớp này bao gồm các class sau: ProductBL, CustomerBL, CategoryBL, SaleRetailStoreBL, StatisticBL, UserAccountBL, Common.
* Data Access Layer: lớp này có nhiệm vụ xử lý các yêu cầu thao tác lên dữ liệu như cập nhật CSDL, đọc cơ sở dự liệu và trả về cho lớp Business Logic. Lớp này bao gồm các class như: UserDB, ProductDB, CustomerDB, CategoryDB, SaleRetailStoreDB.
* Data Transfer Object: Lớp này sẽ tạo ra các lớp object để hỗ trợ cho Data Business Logic Layer xử lý và sử dụng trong việc truyền tham số xuống lớp Data Access. Lớp này bao gồm các class như: ProductDTO, CategoryDTO, CustomerDTO, SaleRetailStoreDTO, UserDTO,.

1. **Glossary of terms:**
2. **Other information:**