**Đề bài: Hệ thống quản lý, mua bán hàng trực tuyến của cửa hàng thể thao với yêu cầu:**

**Về phía cửa hàng(admin):** Đăng nhập với tên tài khoản “admin”

+ **admin** có quyền **thêm tài khoản khách hàng**(UserId,Username, password, Full Name, Email, Phone Number, Address, Gender, Date Of Birth), **sửa tài khoản khách hàng** (password, Full Name, Email, Phone Number, Address, Gender, Date Of Birth), **xoá tài khoản khách hàng** với UserId tự tăng dần theo mẫu “KHXXXXXX” (XXXXXX tăng dần từ 1), UserName phải là các chữ cái hoặc các số và không được bắt đâu bằng cụm “admin”, password phải từ 8 ký tự trở lên bao gồm chữ cái và chữ số, Phone Number chỉ có 10 số , thêm tài khoản phải thêm giỏ hàng cho người đó và phải thêm đầy đủ thông tin, sửa tài khoản dựa trên userName, xoá tài khoản dựa trên userName

+ **admin** có quyền **thêm sản phẩm mới vào kho cửa hàng** (ProductId, name, description, price, quantity In Stock, category, brand, status), **sửa thông tin sản phẩm**(name, description, price, quantity In Stock, category, brand, status), **xoá sản phẩm khỏi kho hàng**, **tìm kiếm sản phẩm** trong kho theo name với ProductId tự tăng dần theo mẫu “SPXXXXXX” (XXXXXX tăng dần từ 1), giá trị Price là số thực, quantityInStock là số nguyên dương, Status chỉ có 3 sự lựa chọn là “In Stock, Out Stock, Supension of sales”, Thêm sản phẩm phải điền đầy đủ các thông tin của sản phẩm đó, Sửa thông tin sản phẩm dựa trên productId, Xoá sản phẩm khỏi kho hàng dựa trên productId

+**admin** có quyền **đổi mật khẩu của mình** đáp ứng mật khẩu từ 8 ký tự trở lên gồm chữ và số

**Về phía khách hàng(user):** Đăng nhập với tài khoản mà user đó đã đăng ký  
 +**user thêm sản phẩm vào giỏ hàng** của mình, **tìm kiếm sản phẩm** trong kho theo name, chú ý số lượng thêm vào giỏ phải nhỏ hơn hoặc bằng số lượng trong kho hàng đang có, khách hàng chỉ được thêm sản phẩm vào giỏ hàng khi trạng thái(status) của sản phẩm là “In Stock”

+**user** có thể **xem thông tin các sản phẩm đã thêm giỏ hàng** của mình(productId, name, Price, Quantity, Total) và tổng giá trị của giỏ hàng

+**user** chỉnh **sửa số lượng của sản phẩm** **nào đó trong giỏ hàng** với số lượng thay đổi trong giỏ phải nhỏ hơn hoặc bằng số lượng trong kho hàng đang có

+**user** có thể **xoá sản phẩm ra khỏi giỏ hàng của mình**

+**user** có thể **đặt hàng** với các ràng buộc số lượng trong giỏ hàng phải nhỏ hơn hoặc bằng số lượng trong kho hàng đang có và sản phẩm vào giỏ hàng khi trạng thái(status) của sản phẩm là “In Stock”

+**user** có thể **thay đổi các thông tin** của mình(password, Full Name, Email, Phone Number, Address, Gender, Date Of Birth) password phải từ 8 ký tự trở lên bao gồm chữ cái và chữ số, Phone Number chỉ có 10 số

# I. Phân tích thiết kế các lớp:

1. Lớp MySQLConnector:

Thuộc tính:

* dataBaseURL: String
* username: String
* password: String

Phương thức:

* getConnection(): Connection

2. Lớp Product:

Thuộc tính:

* productId: String
* name: String
* description: String
* price: double
* quantityInStock: int
* category: String
* brand: String
* status: String

Phương thức:

* Product (name: String, description: String, price: double, quantityInStock: int, category: String, brand: String, status: String): void
* Product (productId: String, name: String, description: String, price: double, quantityInStock: int, category: String, brand: String, status: String): void
* Product (productId: String, name: String, price: double, quantityInStock: int): void
* Product (): void
* getter-setter

3. Lớp ProductDAO:

Thuộc tính:

Phương thức:

* ProductDAO(): void
* addProduct (product: Product): boolean
* getQuantityInStock (productId: String): int
* readProduct (name: String): Product
* readProductId (productId: String): Product
* updateQuantity (product: Product): boolean
* updateProduct (product: Product): boolean
* deleteProduct (productId: String): boolean
* readProductsTableData (): String[][]
* searchProducts (keyword: String): String[][]

4. Lớp User:

Thuộc tính:

* userId: String
* UserName: String
* PassWord: String
* fullName: String
* email: String
* phoneNumber: String
* address: String
* gender: String
* dateOfBirth: String

Phương thức:

* User (userName: String, password: String, fullName: String, email: String, phoneNumber: String, address: String, gender: String, dateOfBirth: String): void
* User (userId: String, userName: String, password: String, fullName: String, email: String, phoneNumber: String, address: String, gender: String, dateOfBirth: String): void
* getter- setter

5. Lớp UserDAO:

Thuộc tính:

Phương thức:

* UserDAO (): void
* addUser (user: User): boolean
* readUser (username: String): User
* updateUser (user: User): boolean
* deleteUser (username: String): boolean
* readAccountsTableData (): String[][]

6. Lớp Cart:

Thuộc tính:

* cartId: String
* userId: String
* totalPrice: double

Phương thức:

* Cart(cartId: String, userId: String, totalPrice: double): void
* Cart (): void
* getter- setter

7. Lớp CartItemSQL:

Thuộc tính:

* productId: String
* name: String
* price: double
* quantity: int
* total: double

Phương thức:

* CartItemSQL (productId: String, name: String, price: double, quantity: int, total: double): void
* getter- setter

8. Lớp CartDAO:

Thuộc tính:

Phương thức:

* CartDAO(): void
* addCart(Cart cart): boolean
* getCartByUserId(String userId): Cart
* calculateTotalPrice(String cartId): double
* updateTotalPrice(String cartId, double totalPrice): boolean
* addProductToCart(String cartId, String productId, int quantity): boolean
* isProductInCart(String cartId, String productId): boolean
* updateProductQuantity(String userId, String productId, int quantity): boolean
* getProductQuantityInCart(String cartId, String productId): int
* deleteProductCart(String userId, String productId): boolean
* clearCart(String userId): boolean
* readCartTableData(String cartId): String[][]

9. Lớp Account:

Thuộc tính:

* userName: String
* password: String

Phương thức:

* Account(userName: String, password: String): void
* getter

10. Lớp AdminDAO:

Thuộc tính:

Phương thức:

* readAdmin(userName: String): Account
* updatePassword(userName: String, password: String)

# II. Biểu đồ lớp:

|  |  |
| --- | --- |
| MySQLConnector | |
| - dataBaseURL: String | URL của cơ sở dữ liệu MySQL |
| - username: String | Tên người dùng để đăng nhập vào MySQL. |
| - password: String | Mật khẩu |
|  |  |
| + getConnection(): Connection | Thiết lập kết nối với cơ sở dữ liệu MySQL |

|  |  |
| --- | --- |
| Product | |
| - productId: String | ID của sản phẩm |
| - name: String | Tên sản phẩm |
| - description: String | Mô tả sản phẩm |
| - price: double | Giá sản phẩm |
| - quantityInstock: int | Số lượng sản phẩm có sẵn |
| - category: String | Danh mục sản phẩm |
| - brand: String | Thương hiệu sản phẩm |
| - status: String | Trạng thái sản phẩm |
|  |  |
| + Product (name: String, description: String, price: double, quantityInStock: int, category: String, brand: String, status: String): void | Khởi tạo |
| + Product (productId: String, name: String, description: String, price: double, quantityInStock: int, category: String, brand: String, status: String): void | Khởi tạo |
| + Product (): void | Khởi tạo |
| + getter | Truy cập đến giá trị pravate |
| + setter | Cập nhật lại giá trị pravate |

|  |  |
| --- | --- |
| ProductDAO | |
| + ProductDAO (): void | Khởi tạo |
| + addProduct (product: Product): boolean | Thêm sản phẩm vào kho hàng |
| + getQuantityInStock (productId: String): int | Đêm số lượng của sản phẩm |
| + readProduct (name: String): Product | Đọc thông tin sản phẩm |
| + readProductId (productId: String): Product | Đọc thông tin sản phẩm |
| + updateQuantity (product: Product): boolean | Cập nhật số lượng sản phẩm |
| + updateProduct (product: Product): boolean | Cập nhật thông tin sản phẩm |
| + deleteProduct (productId: String): boolean | Xoá sản phẩm |
| + readProductsTableData (): String[][] | Đọc sản phẩm |
| + searchProducts (keyword: String): String[][] | Tìm kiếm sản phẩm dựa vào tên sản phẩm |

|  |  |
| --- | --- |
| User | |
| - userId: String | Id khách hàng |
| - userName | Tài khoản khách hàng |
| - passWord | Mật khẩu |
| - fullName: String | Họ và tên |
| - email: String | Địa chỉ email |
| - phoneNumber: String | Số điện thoại |
| - address: String | Địa chỉ |
| - gender: String | Giới tính |
| - dateOfBirth: String | Năm sinh |
|  |  |
| + User: (userName: String, password: String, fullName: String, email: String, phoneNumber: String, address: String, gender: String, dateOfBirth: String) | Khởi tạo |
| + User: (userId: String, userName: String, password: String, fullName: String, email: String, phoneNumber: String, address: String, gender: String, dateOfBirth: String) | Khởi tạo |
| + getter | Truy cập đến giá trị pravate |
| + setter | Cập nhật lại giá trị pravate |

|  |  |
| --- | --- |
| UserDAO | |
| + UserDAO(): void | Khởi tạo |
| + addUser (user: User): boolean | Thêm tài khoản khách hàng |
| + readUser (username: String): User | Đọc tài khoản khách hàng |
| + updateUser (user: User): boolean | Cập nhật |
| + deleteUser (username: String): boolean | Xoá tài khoản khách hàng |
| + readAccountsTableData (): String[][] | Đọc tài khoản khách hàng |

|  |  |
| --- | --- |
| Cart | |
| - cartId: String | Id giỏ hàng |
| - userId: String | id Khách hàng |
| - totalPrice: double | Tổng giá giỏ hàng |
| + Cart(cartId: String, userId: String, totalPrice: double): void | Khởi tạo |
| + Cart (): void | Khởi tạo |
| + getter | Thêm tài khoản khách hàng |
| + setter | Đọc tài khoản khách hàng |

|  |  |
| --- | --- |
| CartitemSQL | |
| - productId: String | Id sản phẩm |
| - name: String | Tên sản phẩm |
| - price: double | Giá sản phẩm |
| - quantity: int | Số lượng |
| - total: double | Giá của các sản phẩm |
| + CartItemSQL (productId: String, name: String, price: double, quantity: int, total: double): void | Khởi tạo |
| + getter | Truy cập đến giá trị pravate |
| + setter | Cập nhật lại giá trị pravate |

|  |  |
| --- | --- |
| CartDAO | |
| + CartDAO(): void | Khởi tạo |
| + addCart(Cart cart): boolean | Tạo giỏ hàng |
| + getCartByUserId(String userId): Cart | Tìm giỏ hàng |
| + calculateTotalPrice(String cartId): double | Tính tổng giá trị giỏ hàng từ bảng CartItems và Product |
| + updateTotalPrice(String cartId, double totalPrice): boolean | Cập nhật tổng giá trị giỏ hàng sau khi tính toán |
| + addProductToCart(String cartId, String productId, int quantity): boolean | Thêm sản phẩm vào giỏ hàng |
| + isProductInCart(String cartId, String productId): boolean | Kiểm tra xem sản phẩm đã tồn tại trong giỏ hàng chưa |
| + updateProductQuantity(String userId, String productId, int quantity): boolean | Cập nhật số lượng sản phẩm trong giỏ hàng |
| + getProductQuantityInCart(String cartId, String productId): int | Đếm số lượng của sản phẩm x trong giỏ hàng |
| + deleteProductCart(String userId, String productId): boolean | Xoá sản phẩm khỏi giỏ hàng |
| + clearCart(String userId): boolean | Xoá toàn bộ sản phẩm khỏi giỏ hàng |
| + readCartTableData(String cartId): String[][] | Đọc giỏ hàng |

|  |  |
| --- | --- |
| Account | |
| - userName: String | Tên tài khoản |
| - password: String | Mật khẩu |
| + Account(userName: String, password: String): void | Khởi tạo |
| + getter | Truy cập đến giá trị pravate |

|  |  |
| --- | --- |
| AdminDAO | |
| + readAdmin(userName: String): Account | Đọc admin |
| + updatePassword(userName: String, password: String): boolean | Cập nhật mật khẩu |

**III Quan hệ giữa các lớp:**

1. Lớp MySQLConnector

Mục đích: Kết nối cơ sở dữ liệu.

Quan hệ:

* Được sử dụng bởi các lớp DAO: ProductDAO, UserDAO, CartDAO, AdminDAO.

2. Lớp Product

Mục đích: Đại diện cho thông tin sản phẩm.

Quan hệ:

* 1 ProductDAO quản lý nhiều Product.
* 1 Cart quản lý nhiều product

3. Lớp ProductDAO

Mục đích: Thao tác dữ liệu sản phẩm trong cơ sở dữ liệu.

Quan hệ:

* Sử dụng: MySQLConnector để kết nối cơ sở dữ liệu.
* Quản lý: Nhiều đối tượng Product.

4. Lớp User

Mục đích: Đại diện thông tin tài khoản khách hàng.

Quan hệ:

* 1 UserDAO quản lý nhiều User.
* 1 User có 1 Cart.

5. Lớp UserDAO

Mục đích: Thao tác dữ liệu người dùng trong cơ sở dữ liệu.

Quan hệ:

* Sử dụng: MySQLConnector để kết nối cơ sở dữ liệu.
* Quản lý: Nhiều đối tượng User.

6. Lớp Cart

Mục đích: Đại diện cho giỏ hàng của khách hàng.

Quan hệ:

* 1 User có 1 Cart.
* 1 CartDAO quản lý nhiều Cart.
* 1 Cart chứa nhiều CartItemSQL.

7. Lớp CartItemSQL

Mục đích: Đại diện cho từng sản phẩm trong giỏ hàng.

Quan hệ:

* 1 Cart chứa nhiều CartItemSQL.

8. Lớp CartDAO

Mục đích: Quản lý dữ liệu giỏ hàng.

Quan hệ:

* Sử dụng: MySQLConnector để kết nối cơ sở dữ liệu.
* Quản lý: Nhiều Cart và CartItemSQL.

9. Lớp Account

Mục đích: Đại diện thông tin đăng nhập của admin.

Quan hệ:

* 1 AdminDAO quản lý nhiều User.

10. Lớp AdminDAO

Mục đích: Quản lý thông tin tài khoản admin.

Quan hệ:

* Sử dụng: MySQLConnector để kết nối cơ sở dữ liệu.
* Quản lý: Đối tượng Account.