# Tài liệu Thiết kế Cấp thấp (LLD)

# Low\_Level\_Design.docx

\*\*Ngày:\*\* 20/10/2023

\*\*Tác giả:\*\* AI Assistant

# Low-Level Design: Hệ thống Quản Lý Kho Hàng - Mô-đun Quản lý Hàng Hóa và Mô-đun Quản lý Nhập/Xuất Kho

Tài liệu này trình bày thiết kế cấp thấp (Low-Level Design - LLD) cho hai mô-đun quan trọng của Hệ thống Quản Lý Kho Hàng: Mô-đun Quản lý Hàng Hóa và Mô-đun Quản lý Nhập/Xuất Kho. Thiết kế này dựa trên High-Level Design (HLD) đã được xây dựng trước đó và giả định rằng tài liệu SRS đầy đủ đã được cung cấp.

## 1. Mô-đun Quản lý Hàng Hóa

### 1.1. Tổng quan

Mô-đun này cho phép người dùng quản lý thông tin về các mặt hàng trong kho, bao gồm thêm, sửa, xóa và tìm kiếm thông tin hàng hóa.

### 1.2. Cấu trúc lớp (Class Diagram)

```plantuml  
@startuml  
class Product {  
 - productId: Long  
 - productName: String  
 - productCode: String  
 - unitPrice: Double  
 - quantityInStock: Integer  
 - supplierId: Long  
 + getProductDetails(): ProductDetails  
 + updateProductDetails(ProductDetails): void  
}

class ProductDetails {  
 - description: String  
 - unit: String  
 - imageUrl: String  
 + getDescription(): String  
 + getUnit(): String  
 + getImageUrl(): String  
}

class Supplier {  
 - supplierId: Long  
 - supplierName: String  
 - contactPerson: String  
 - contactNumber: String  
 + getSupplierDetails(): SupplierDetails  
}

class SupplierDetails {  
 - address: String  
 - email: String  
 + getAddress(): String  
 + getEmail(): String  
}

Product "1" -- "1" Supplier : belongs to

@enduml  
```

### 1.3. Phương thức (Methods)

\*\*Lớp `Product`:\*\*

\* `getProductDetails()`: Trả về đối tượng `ProductDetails` chứa thông tin chi tiết của sản phẩm.  
\* `updateProductDetails(ProductDetails)`: Cập nhật thông tin chi tiết của sản phẩm.

\*\*Lớp `ProductDetails`:\*\*

\* `getDescription()`, `getUnit()`, `getImageUrl()`: Trả về thông tin mô tả, đơn vị tính và URL hình ảnh của sản phẩm.

\*\*Lớp `Supplier`:\*\*

\* `getSupplierDetails()`: Trả về đối tượng `SupplierDetails` chứa thông tin chi tiết của nhà cung cấp.

\*\*Lớp `SupplierDetails`:\*\*

\* `getAddress()`, `getEmail()`: Trả về địa chỉ và email của nhà cung cấp.

### 1.4. Thuật toán và Cấu trúc dữ liệu

\* Dữ liệu sản phẩm được lưu trữ trong cơ sở dữ liệu quan hệ (PostgreSQL hoặc MySQL).  
\* Sử dụng ORM (Object-Relational Mapping) để ánh xạ đối tượng Java/Go với bảng trong cơ sở dữ liệu.  
\* Tìm kiếm sản phẩm có thể sử dụng chỉ mục cơ sở dữ liệu để tăng hiệu suất.

## 2. Mô-đun Quản lý Nhập/Xuất Kho

### 2.1. Tổng quan

Mô-đun này quản lý các giao dịch nhập/xuất kho, cập nhật số lượng hàng tồn kho và tạo các báo cáo liên quan.

### 2.2. Cấu trúc lớp (Class Diagram)

```plantuml  
@startuml  
class Transaction {  
 - transactionId: Long  
 - productID: Long  
 - transactionType: String // "IN" or "OUT"  
 - quantity: Integer  
 - timestamp: Date  
 - userId: Long  
 + getTransactionDetails(): TransactionDetails  
}

class TransactionDetails {  
 - description: String  
 + getDescription(): String  
}

@enduml  
```

### 2.3. Phương thức (Methods)

\*\*Lớp `Transaction`:\*\*

\* `getTransactionDetails()`: Trả về đối tượng `TransactionDetails` chứa thông tin chi tiết của giao dịch.

### 2.4. Thuật toán và Cấu trúc dữ liệu

\* Mỗi giao dịch được lưu trữ như một bản ghi trong bảng `transactions` trong cơ sở dữ liệu.  
\* Sử dụng trigger hoặc stored procedure trong cơ sở dữ liệu để cập nhật số lượng hàng tồn kho sau mỗi giao dịch.  
\* Có thể sử dụng message queue (như RabbitMQ hoặc Kafka) để xử lý các giao dịch đồng thời và đảm bảo tính nhất quán dữ liệu.

## 3. Giao tiếp giữa các Mô-đun

Mô-đun Quản lý Hàng Hóa và Mô-đun Quản lý Nhập/Xuất Kho sẽ giao tiếp với nhau thông qua API RESTful. Ví dụ, khi một giao dịch nhập/xuất kho được tạo, Mô-đun Quản lý Nhập/Xuất Kho sẽ gọi API của Mô-đun Quản lý Hàng Hóa để cập nhật số lượng hàng tồn kho của sản phẩm tương ứng.

## 4. Xử lý lỗi và Logging

Mỗi mô-đun sẽ có cơ chế xử lý lỗi riêng, bao gồm logging chi tiết các lỗi và ngoại lệ. Thông tin log sẽ được lưu trữ trong một hệ thống log tập trung (ví dụ: Elasticsearch, Logstash, Kibana).

## 5. Kiểm thử

Các mô-đun sẽ được kiểm thử kỹ lưỡng bằng các phương pháp kiểm thử đơn vị, tích hợp và hệ thống.

Lưu ý: Đây là một bản LLD mẫu. Thiết kế chi tiết có thể thay đổi tùy thuộc vào yêu cầu cụ thể của dự án. Các chi tiết về cơ sở dữ liệu, API, và các công nghệ cụ thể sẽ được mô tả chi tiết trong các tài liệu liên quan.