**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**BÁO CÁO BÀI TẬP LỚN**

**NGUYỄN THANH LÂM**

lam.nt20200336@sis.hust.edu.vn

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | TS. Trịnh Tuấn Đạt |
| **Bộ môn:** | Công nghệ phần mềm |
| **Viện:** | Công nghệ thông tin và truyền thông |
| **HÀ NỘI, 12/2023** | |

**MỤC LỤC**

[CHƯƠNG 1. PHÂN TÍCH YÊU CẦU (BÀI 2) 1](#_Toc154529191)

[1.1 Biểu đồ use case 1](#_Toc154529192)

[1.1.1 Biểu đồ use case tổng quan 1](#_Toc154529193)

[1.1.2 Biểu đồ use case phân rã “Xem thông tin chấm công” 3](#_Toc154529194)

[1.1.3 Chỉnh sửa thông tin chấm công 4](#_Toc154529195)

[1.2 Đặc tả Use case 5](#_Toc154529196)

[1.2.1 Use case “Nhập dữ liệu chấm công bằng file excel” 5](#_Toc154529197)

[1.2.2 Biểu đồ hoạt động cho luồng nhập dữ liệu thành công: 7](#_Toc154529198)

[CHƯƠNG 2. THIẾT KẾ USECASE (BÀI 3) 8](#_Toc154529199)

[2.1 Biểu đồ trình tự mức phân tích: 8](#_Toc154529200)

[2.2 Biểu đồ giao tiếp mức phân tích: 8](#_Toc154529201)

[2.3 Biểu đồ lớp mức phân tích: 9](#_Toc154529202)

[CHƯƠNG 3. THIẾT KẾ GIAO DIỆN (BÀI 4) 10](#_Toc154529203)

[3.1 Sơ đồ chuyển đổi màn hình: 10](#_Toc154529204)

[3.2 Đặc tả giao diện: 10](#_Toc154529205)

[3.2.1 Màn hình giao diện Import (ImportAttendencePanel) 11](#_Toc154529206)

[3.2.2 Màn hình chọn file để import 12](#_Toc154529207)

[3.2.3 Màn hình hiển thị lỗi khi nhập dữ liệu: 13](#_Toc154529208)

[3.2.4 Màn hình nhập dữ liệu thất bại: 14](#_Toc154529209)

[3.3 Biểu đồ lớp cho Subsystem 15](#_Toc154529210)

[CHƯƠNG 4. THIẾT KẾ CHI TIẾT LỚP (BÀI 5) 16](#_Toc154529211)

[4.1 Biểu đồ trình tự mức thiết kế 16](#_Toc154529212)

[4.1.1 Scenerio: Import thành công 16](#_Toc154529213)

[4.1.2 Scenerio: Import thất bại do file không đúng định dạng dữ liệu 16](#_Toc154529214)

[4.1.3 Scenerio: Import thất bại do trùng dữ liệu 17](#_Toc154529215)

[4.2 Biểu đồ lớp mức thiết kế 17](#_Toc154529216)

[4.3 Sybsystem: Hệ thống quản lý nhân sự 18](#_Toc154529217)

[4.4 Gói 19](#_Toc154529218)

[CHƯƠNG 5. CẢI THIỆN BIỂU ĐỒ TRÌNH TỰ VÀ LỚP MỨC THIẾT KẾ (BÀI 6) 21](#_Toc154529219)

[5.1 Cách chỉnh sửa và ý nghĩa mang lại 21](#_Toc154529220)

[5.1.1 Áp dụng mẫu thiết kế Singleton 21](#_Toc154529221)

[5.1.2 Áp dụng mẫu thiết kế Factory 21](#_Toc154529222)

[5.1.3 Áp dụng nguyên tắc Cohesion and Coupling 21](#_Toc154529223)

[5.1.4 Áp dụng nguyên lý SOLID 21](#_Toc154529224)

[CHƯƠNG 6. LẬP TRÌNH VÀ KIỂM THỬ ĐƠN VỊ 23](#_Toc154529225)

[6.1 Thiết kế test case cho module: ImportService 23](#_Toc154529226)

[6.1.1 Mô tả module: 23](#_Toc154529227)

[6.1.2 Thiết kế test case cho phương thức ImportOfficerAttendace: 23](#_Toc154529228)

[6.1.3 Thiết kế test case cho phương thức createSession: 26](#_Toc154529229)

[6.1.4 Thiết kế test case cho usecase: Nhập dữ liệu bằng file excel 28](#_Toc154529230)

# PHÂN TÍCH YÊU CẦU (BÀI 2)

## Biểu đồ use case

### Biểu đồ use case tổng quan

A diagram of people with text

Description automatically generated

Giải thích về các tác nhân:

* **Nhân viên**: Người dùng cấp thấp nhất, chỉ có thể xem được các thông tin chấm công liên quan đến mình
* **Nhân viên văn phòng**: Kế thừa nhân viên, chỉ xem được thông tin chấm công dạng nhân viên văn phòng.
* **Công nhân**: Kế thừa nhân viên, chỉ xem được thông tin chấm công dạng công nhân.
* **Trưởng đơn vị**: Người dùng cấp cao hơn nhân viên, có thể xem được thông tin chấm công của các thành viên trong đơn vị của mình quản lý.
* **Quản lý nhân sự**: Người dùng cấp cao nhất, có thể xem được thông tin chấm công của toàn bộ nhân viên công ty, có quyền chỉnh sửa thông tin chấm công, xem báo cáo, import/export dữ liệu chấm công.
* **Máy quét vân tay**: Hệ thống bên ngoài có nhiệm vụ cung cấp API cho hệ thống lấy dữ liệu chấm công
* **Hệ thống quản lý nhân sự**: Hệ thống bên ngoài có nhiệm vụ cung cấp API cho hệ thống chấm công lấy thông tin của nhân viên.
* **Hệ thống kế toán**: Hệ thống bên ngoài có khả năng lấy thông tin chấm công từ hệ thống chấm công để tính lương của nhân viên cho từng tháng.

Giải thích về các use case:

Composite use case:

* **Xem thông tin chấm công**: Use case tổng quát của các use case thực hiện các hành vi nghiệp vụ lên máy chấm công vân tay.
* **Chỉnh sửa thông tin chấm công**: Use case tổng quát của các use case thực hiện các hành vi nghiệp vụ lên

Actual use case:

* **Cung cấp dữ liệu chấm công nhân viên qua API**
* **Đăng kí tài khoản**
* **Đăng nhập**
* **Lấy dữ liệu chấm công nhân viên qua API**

### A diagram of a diagram Description automatically generatedBiểu đồ use case phân rã “Xem thông tin chấm công”

### Chỉnh sửa thông tin chấm công

A diagram of a person's diagram

Description automatically generated

## Đặc tả Use case

### Use case “Nhập dữ liệu chấm công bằng file excel”

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Use case | UC002 | Tên Use case | Nhập dữ liệu chấm công vân tay qua excel |
| Tác nhân | Người quản lý nhân sự | | |
| Tiền điều kiện | Quản lý nhân sự đã đăng nhập vào hệ thống và đang ở trên giao diện “Nhập dữ liệu chấm công bằng excel” | | |
| Luồng sự kiện chính  (Thành công) | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1 | Người quản lý nhân sự | Click vào nút “Chọn file chấm công” | | 2 | Hệ thống | Hiển thị hộp thoại để người dùng có thể chọn file excel để import | | 3 | Người quản lý nhân sự | Chọn file cần import từ danh sách các file ở USB | | 4 | Hệ thống | Phân tích nội dung của file excel để trích xuất thông tin về thời gian (timestamp) và mã nhân viên, sau đó hiển thị lên bảng dữ liệu | | 5 | Người quản lý nhân sự | Click vào nút “Nhập dữ liệu” | | 6 | Hệ thống | Hiển thị confirm box để lựa chọn xác nhận hoặc đóng | | 7 | Người quản lý nhân sự | Click vào nút “Đồng ý” | | 8 | Hệ thống | Kiểm tra tính hợp lệ của dữ liệu trước khi import (đúng định dạng, không trùng lặp, … ) | | 9 | Hệ thống | Bắt đầu import dữ liệu vào cơ sở dữ liệu | | 10 | Hệ thống | Ghi nhận lại việc import này để có thể sau này cần đối chiếu | | 11 | Hệ thống | Thông báo nhập dữ liệu thành công | | | |
| Luồng sự kiện thay thế | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 4.a | Hệ thống | Hiển thị thông báo lỗi: Sai định dạng file | | 5.a | Người quản lý nhân sự | Nhấn nút “Đóng”. Hệ thống trở lại giao diện xem lịch sử import dữ liệu | | 6.a | Hệ thống | Thông báo lỗi: thông báo lỗi dữ liệu về định dạng, thời gian, trùng nhau, ... | | 7.a | Người quản lý nhân sự | Nhấn nút “Hủy”, hệ thống tắt comfirm box và không làm gì | | 8.a | Hệ thống | Thông báo lỗi dữ liệu trong file | | 9.a | Hệ thống | Thông báo dữ liệu bị trùng lặp. Rollback lại trạng thái cũ | | | |
| Hậu điều kiện | Không | | |

\* Dữ liệu đầu vào:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Trường dữ liệu** | **Mô tả** | **Bắt buộc?** | **Điều kiện hợp lệ** | **Ví dụ** |
| 1 | File excel | File chứa thông tin thời gian và mã nhân viên | Có | Đúng định dạng file, có các cột trường thông tin thời gian và mã nhân viên, các cell data đúng về định dạng (ngày, mã) | Chamcong\_1\_2023.xlsx |

\* Dữ liệu đầu vào của file excel gồm:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Bắt buộc | Điều kiện hợp lệ | Ví dụ |
| 1 | Timestamp | Thời gian chấm công | Có | Đúng định dạng ngày tháng năm | 08:02:14 04/10/2023 |
| 2 | Mã nhân viên | Mã nhân viên chấm công | Có | Định dạng chuỗi | NV123456 |

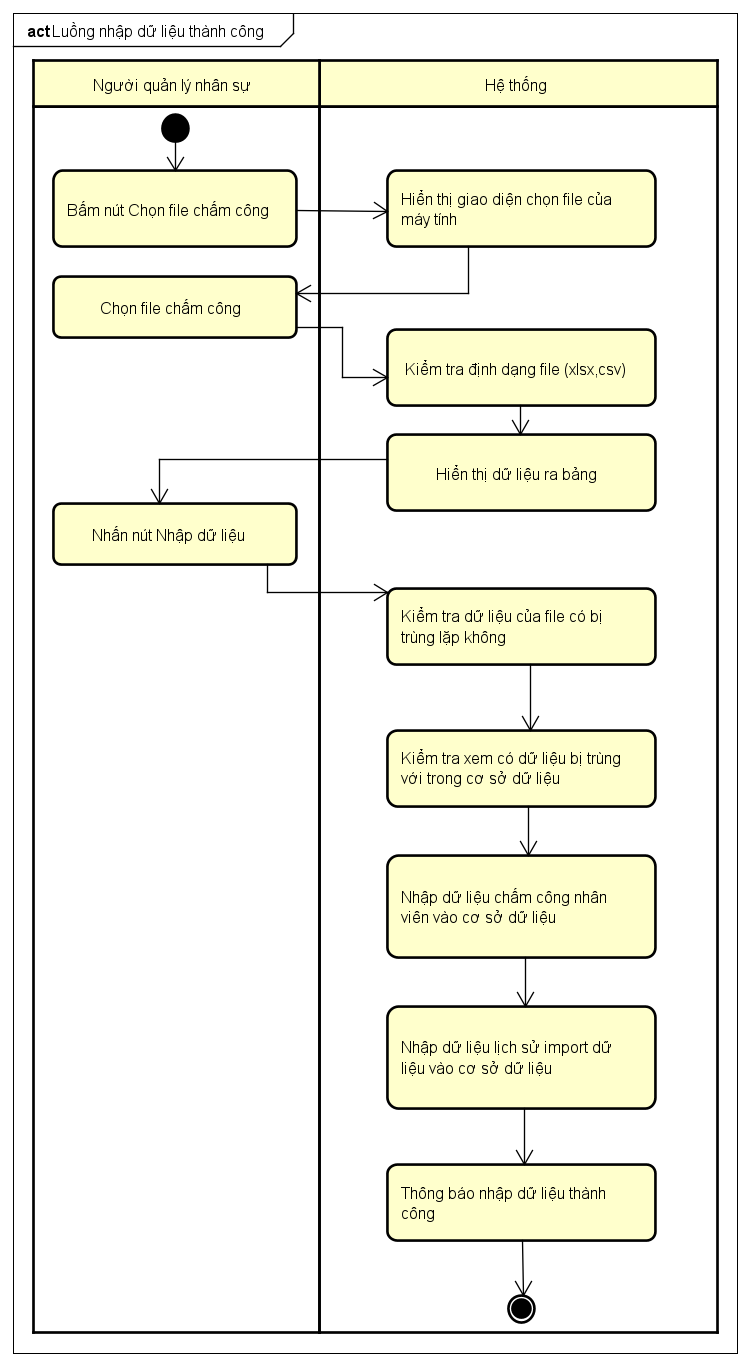
\* Dữ liệu đầu ra của công nhân hiển thị:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Worker | **Ngày** | **Thứ 2 1/2/2023** | **Thứ 3 1/2/2023** | **Thứ 4**  **3/2/2021** | **…** |
| 20170534  (mã nhân  viên) | Ca 1 (shift1) | 4.0 | 4.0 | 4.0 |  |
| Ca 2 (shift2) | 4.0 | 3.5 | 4.0 |  |
| Ca 3 (shift3) | 0.0 | 1.0 | 0.0 |  |

\* Dữ liệu đầu ra của nhân viên văn phòng hiển thị:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Officer | **Ngày** | **Thứ 2 1/2/2021** | **Thứ 3 2/2/2021** | **Thứ 4 3/2/2021** | **…** |
| 20170535 (mã nhân viên) | Sáng (morningSession) | có | có | Có |  |
| Chiều (afternoonSession) | Không | Có | Có |  |
| Đi muộn (hoursLate) | 0.25 | 0 | 0 |  |
| Về sớm (hoursEarlyLeave) | 0.1 | 0 | 0 |  |

### Biểu đồ hoạt động cho luồng nhập dữ liệu thành công:



# THIẾT KẾ MỨC PHÂN TÍCH (BÀI 3)

## Biểu đồ trình tự mức phân tích:

A diagram of a project

Description automatically generated

## Biểu đồ giao tiếp mức phân tích:

A diagram of a diagram

Description automatically generated

## Biểu đồ lớp mức phân tích:

A diagram of a work flow

Description automatically generated

# THIẾT KẾ GIAO DIỆN VÀ SUBSYSTEM (BÀI 4)

## Sơ đồ chuyển đổi màn hình:

A diagram of a system

Description automatically generated

A diagram of a diagram

Description automatically generated

## Đặc tả giao diện:

### Màn hình giao diện Import (ImportAttendencePanel)

A screenshot of a computer

Description automatically generated

|  |  |  |
| --- | --- | --- |
| Control | Operation | Function |
| Import Button | Click | Hiện thị ra màn hình import dữ liệu |
| Table | Initial | Hiện ra lịch sử import |
| Paging | Click | Chọn trang dữ liệu hiển thị |

### Màn hình chọn file để import

A screenshot of a computer

Description automatically generated

|  |  |  |
| --- | --- | --- |
| Control | Operation | Function |
| Nút Chọn file chấm công | Click | Hiển thị ra màn hình chọn file |
| Nút X | Click | Đóng màn hình |
| Nút Nhập dữ liệu | Click | Hệ thống bắt đầu import dữ liệu |
| Table | Initial | Hiển thị danh sách dữ liệu lấy được từ file sau khi chọn file |
| Nút Đóng | Click | Hệ thống trở lại màn hình |
| Nút Clear | Click | Clear dữ liệu đang có trên bảng |

### Màn hình hiển thị lỗi khi nhập dữ liệu:

A screenshot of a computer

Description automatically generated

|  |  |  |
| --- | --- | --- |
| Control | Operation | Function |
| Message | Initial | Thông báo |
| Button ‘Đóng’ | Click | Đóng popup thông báo |
| Button ‘X’ | Click | Đóng popup thông báo |

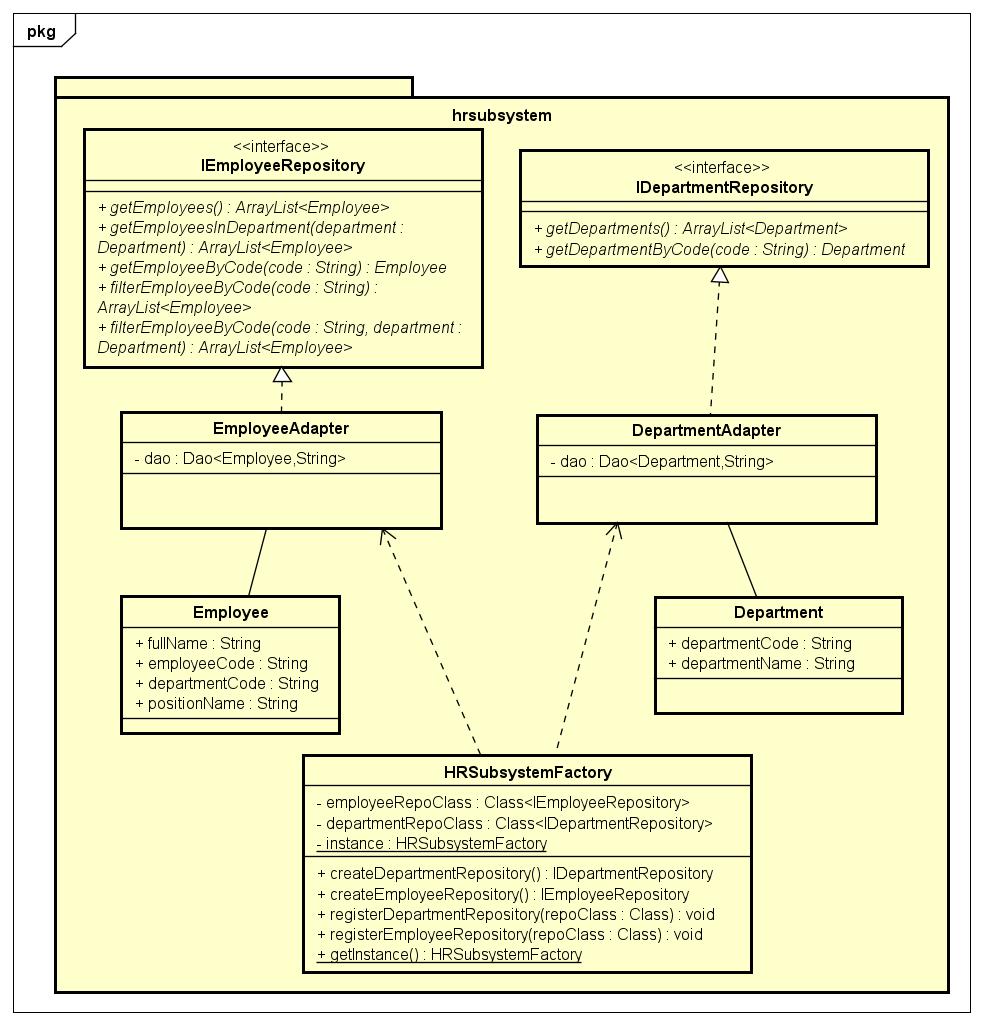
### Màn hình nhập dữ liệu thất bại:

A screenshot of a computer

Description automatically generated

|  |  |  |
| --- | --- | --- |
| Control | Operation | Function |
| Message | Initial | Thông báo |
| Button ‘Đóng’ | Click | Đóng popup thông báo |
| Button ‘X’ | Click | Đóng popup thông báo |

## Biểu đồ lớp cho Subsystem



# BIỂU ĐỒ TRÌNH TỰ VÀ LỚP MỨC THIẾT KẾ (BÀI 5)

## Biểu đồ trình tự mức thiết kế

### Scenerio: Import thành công

A diagram of a project

Description automatically generated with medium confidence

### Scenerio: Import thất bại do file không đúng định dạng dữ liệu

A screenshot of a computer

Description automatically generated

### Scenerio: Import thất bại do trùng dữ liệu

A screenshot of a computer program

Description automatically generated

## Biểu đồ lớp mức thiết kế

A yellow folder with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

## Sybsystem: Hệ thống quản lý nhân sự

A diagram of a company

Description automatically generated

## Gói

A diagram of a diagram

Description automatically generated

# CẢI THIỆN BIỂU ĐỒ TRÌNH TỰ VÀ LỚP MỨC THIẾT KẾ (BÀI 6)

## Cách chỉnh sửa và ý nghĩa mang lại

### Áp dụng mẫu thiết kế Singleton

Mẫu thiết kế Singleton là một trong những mẫu thiết kế phổ biến trong lập trình, được sử dụng trong những lớp**ImportHistoryRepository, ImportServiceFactory, HRSubsystemFactory**

để đảm bảo rằng:

* Một lớp chỉ có duy nhất một thể hiện.
* Có thể dễ dàng truy cập toàn cục đối tượng.
* Không cần phải quan tâm cách khởi tạo đối tượng.

### Áp dụng mẫu thiết kế Factory

Mẫu thiết kế Singleton là một trong những mẫu thiết kế phổ biến trong lập trình, được sử dụng trong những lớp **I ImportHistoryRepository, IimportService, HRSubsystemFactory** được nhằm mục đích:

* **Tạo đối tượng mà không cần biết chi tiết cụ thể**: Mẫu Factory giúp ẩn đi logic cụ thể của việc tạo đối tượng. Khi sử dụng một phương thức tạo đối tượng từ Factory, bạn không cần biết chi tiết cách đối tượng đó được khởi tạo hoặc được xử lý bên trong.
* **Đảm bảo tuân thủ nguyên tắc "Open/Closed"**: Mẫu Factory giúp đảm bảo rằng khi thêm một triển khai mới của interface mà không cần phải sửa đổi mã nguồn hiện tại. Thay vào đó, chỉ cần thêm một lớp mới vào Factory.
* **Tăng sự linh hoạt của hệ thống**: Factory có thể được cấu hình để tạo ra các đối tượng thuộc các lớp con khác nhau dựa trên điều kiện nào đó. Điều này tăng sự linh hoạt và tái sử dụng mã nguồn.
* **Giảm sự phụ thuộc với đối tượng cụ thể**: Khi sử dụng một Factory để tạo đối tượng thì không cần phải biết chi tiết cụ thể cách khởi tạo của đối tượng đó. Điều này giảm sự phụ thuộc và giúp giữ cho mã nguồn linh hoạt hơn.

### Áp dụng nguyên tắc Cohesion and Coupling

Các lớp được phân lại thành các gói **pages, impoirtexcel, attendance, hrsubsystem** với mục đính tăng tính kết dính cho các lớp trong một gói, và giảm sự phụ thuộc giữa các gói với nhau. Những lớp có cùng một nhiệm vụ, cùng một nghiệp vụ thì được gom nhóm lại một gói, và mỗi gói lại có một nhiệm vụ khác nhau.

Cohesion giúp tăng tính đóng gói (encapsulation) và giảm sự phụ thuộc giữa các module. Nó làm cho mã nguồn dễ đọc, dễ bảo trì, và dễ kiểm thử. Cohesion cao cũng giúp tái sử dụng mã nguồn dễ dàng hơn.

Coupling giúp tăng tính linh hoạt và tái sử dụng của hệ thống. Khi các thành phần độc lập và không phụ thuộc quá mức lớn vào nhau, sự thay đổi trong một thành phần không ảnh hưởng quá lớn đến các thành phần khác. Điều này làm cho hệ thống linh hoạt và dễ mở rộng, bảo trì.

### Áp dụng nguyên lý SOLID

* *S - Một class chỉ nên giữ một trách nhiệm duy nhất*

Tất cả các lớp đều chỉ thực hiện một trách nhiệm duy nhất, như các lớp Model **ImportHistory, AttendanceLogImport** chỉ chứa dữ liệu về đối tượng, trong khi các lớp **SqliteImportHistoryRepository** phụ trách việc thao tác lưu trữ đối tượng trong database hoặc tương tác với hệ thống bên ngoài để lấy đối tượng, …

* *O - Có thể thoải mái mở rộng 1 module, nhưng hạn chế sửa đổi bên trong module đó*

Phần lớn các hàm đều được thiết kế theo dạng subcribe, unsubscribe để tránh việc thay đổi logic hàm, như các hàm **createRepository** và **registerRepository** trong lớp **ImportService, ImportServiceFactory, SqlImportHistoryRepository**

* *L - Trong một chương trình, các object của class con có thể thay thế class cha mà không làm thay đổi tính đúng đắn của chương trình*

Không có lớp nào sử dụng kế thừa nên xem như hợp nguyên tắc.

* *I - Thay vì dùng 1 interface lớn, ta nên tách thành nhiều interface nhỏ, với nhiều mục đích cụ thể*

Subsystem được tách thành **IEmployeeRepository, IDepartmentRepository** nhằm để thực hiện truy vấn với từng kiểu dữ liệu lần lượt là **Employee, IDepartment** thay vì dồn vào 1 lớp duy nhất.

* *D - Các module cấp cao không nên phụ thuộc vào các module cấp thấp. Cả 2 nên phụ thuộc vào abstraction. Interface (abstraction) không nên phụ thuộc vào chi tiết, mà ngược lại. (Các class giao tiếp với nhau thông qua interface, không phải thông qua implementation.)*

Thay vì phải để mã phụ thuộc vào **EmployeeAdapter, DepartmentAdapter, SqliteOfficerAtte-ndanceRepository, SqliteUserRepository** thì cho những đoạn mã phụ thuộc thay thế lần lượt vào **IEmployeeRe-pository, IDepartmentRepository, IOfficerAttendanceRepository, IUserRepository** sẽ mang lại những điều sau:

* **Định hình Cấu trúc Hệ thống**: Nguyên tắc Dependency Convention giúp định hình cấu trúc của hệ thống bằng cách quy ước về cách các thành phần phụ thuộc lẫn nhau. Điều này tạo ra một cấu trúc tự nhiên và dễ hiểu, giúp người đọc mã nguồn dễ dàng theo dõi và hiểu cấu trúc tổng thể của hệ thống.
* **Giảm Độ phức tạp**: Nguyên tắc Dependency Convention giúp giảm độ phức tạp của hệ thống bằng cách hạn chế sự phụ thuộc giữa các thành phần (thay vào đó là phụ thuộc vào interface). Điều này giúp giảm nguy cơ xung đột và khó khăn trong quá trình phát triển và bảo trì.
* **Dễ Bảo trì và Nâng cấp**: Nguyên tắc Dependency Convention giúp việc bảo trì và nâng cấp trở nên dễ dàng hơn. Các thay đổi có thể được thực hiện mà không làm ảnh hưởng đến toàn bộ hệ thống, chỉ bằng cách thay đổi implement này bằng implement khác của interface, và người phát triển có thể tập trung vào các thành phần cụ thể mà họ đang làm việc.
* **Tăng Tính tái sử dụng**: Dependency Convention thúc đẩy việc sử dụng lại mã nguồn bằng cách tạo ra các thành phần độc lập, có thể được tái sử dụng trong các phần khác nhau của hệ thống hoặc trong các dự án khác.
* **Tăng Tính Mô đun hóa**: Nguyên tắc này hỗ trợ tính mô đun hóa của hệ thống, trong đó mỗi mô đun (hoặc thành phần) có trách nhiệm và chức năng cụ thể. Điều này giúp tạo ra các mô đun độc lập, dễ kiểm thử và duy trì.

# Lập trình và kiểm thử đơn vị

## Thiết kế test case cho module: ImportService

### Mô tả module:

Module chứa các hàm liên quan tới việc nhập dữ liệu chấm công từ file excel vào cơ sở dữ liệu. Bên trong class này có sử dụng các dependency là các instance của các Class có chức năng làm việc với cơ sở dữ liệu, cung cấp các phương thức để làm các nghiệp vụ của Class ImportService.

### Thiết kế test case cho phương thức ImportOfficerAttendace:

**Tên phương thức**: ImportOfficerAttendace

**Kiểu trả về**: void

**Danh sách tham số**: 1 instance của class File của module java.io.file trong java

**Chi tiết**: Hàm nhận vào 1 file của máy tính và thực hiện import dữ liệu trong file vào database. File chứa thông tin các log chấm công lấy từ máy chấm công và định dạng phù hợp là .xlsx.

#### Kỹ thuật sử dụng kiểm thử hộp đen và phân vùng tương đương.

Kiểm thử hộp đen: Không cần biết logic bên trong code, chỉ cần biết định dạng File hợp lệ và không hợp lệ, kết hợp với giao diện người dùng để kiểm thử.

Phân vùng tương đương cho phép phân chia tập hợp các điều kiện kiểm tra thành một phân vùng nên được coi là giống nhau. Để thiết kế được các test case cho phương thức này, đầu tiên xác định các vùng tương đương của đầu vào, ở đây là File excel.

**Vùng tương đương hợp lệ:**

+ File có 2 cột giá trị. Cột đầu tiên chứa giá trị các mã nhân viên, cột thứ 2 chứa giá trị thời gian. Giá trị mã nhân viên phải có trong database và định dạng thời gian phải là dd/MM/yyyy hh:mm:ss.

**Vùng tương đương không hợp lệ**:

+ File không đúng định dạng excel.

+ File không có 2 cột giá trị.

+ Có 2 dòng giá trị trùng nhau.

+ Có mã nhân viên không tồn tại trong cơ sở dữ liệu.

+ Có thời gian không đúng định dạng dd/MM/yyyy hh:mm:ss.

+ Có dòng dữ liệu trùng với dữ liệu đã có trong database.

Tên class kiểm thử: ImportDataExcelTest. Các phương thức kiểm thử trong class:

1. ImportDataExcel\_Success: test trường hợp thành công khi chọn file dữ liệu thỏa mãn.
2. ImportDataExcel\_InvalidFileExtension: test trường hợp định dạng file không đúng.
3. ImportDataExcel\_NullRow: test trường hợp file không có 2 cột dữ liệu.
4. ImportDataExcel\_DuplicateRow: test trường hợp file có 2 dòng dữ liệu giống nhau.
5. ImportDataExcel\_ExistRow: test trường hợp file có dòng dữ liệu đã tồn tại trong cơ sở dữ liệu.
6. ImportDataExcel\_CodeNotExist: test trường hợp có mã không tồn tại trong cơ sở dữ liệu.
7. ImportDataExcel\_InvalidTime: test trường hợp có thời gian không đúng định dạng.

#### Thiết kế test case.

**Test case ImportDataExcel\_Success**

|  |  |
| --- | --- |
| **Test case #:** 4.1 | **Tên test case:** ImportDataExcel |
| **Hệ thống:** Hệ thống quản lý chấm công | **Hệ thống con:** Hệ thống quản lý nhân sự |
| **Thiết kế bởi:** Nguyễn Thanh Lâm 20200336 | **Ngày thiết kế:** 20/12/2023 |
| **Thực thi bởi:** ImportService | **Ngày thực thi:** 24/12/2023 |
| **Mô tả ngắn gọn:** Kiểm tra chức năng import dữ liệu chấm công bằng file excel của phương thức ImportOfficerAttendace thành công |  |

|  |
| --- |
| **Tiền điều kiện:**  Hệ thống đang hiển thị giao diện chọn file để import  File có định dạng và dữ liệu hợp lệ |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bước | Hành động | Phản hồi mong muốn từ hệ thống | Pass/Fail | Comment |
| 1 | Người dùng click nút chọn file import | Hệ thống hiển thị giao diện chọn file | Pass |  |
| 2 | Người dùng chọn file | Hệ thống hiển thị giá trị các cột của file lên 1 bảng. | Pass |  |
| 3 | Người dùng chọn nút Nhập dữ liệu | Hệ thống hiển thị Confirm box để xác nhận lần nữa | Pass |  |
| 4 | Người dùng ấn nút Đồng ý | Hệ thống thông báo import thành công | Pass |  |

|  |
| --- |
| **Trạng thái của hệ thống sau khi chạy hàm:**  Hệ thống nhập dữ liệu về lịch sử import và các thông tin về ca chấm công từ file vào trong database. |

**Test case ImportDataExcel\_InvalidData**

|  |  |
| --- | --- |
| **Test case #:** 4.2 | **Tên test case:** ImportDataExcel |
| **Hệ thống:** Hệ thống quản lý chấm công | **Hệ thống con:** Hệ thống quản lý nhân sự |
| **Thiết kế bởi:** Nguyễn Thanh Lâm 20200336 | **Ngày thiết kế:** 20/12/2023 |
| **Thực thi bởi:** ImportService | **Ngày thực thi:** 24/12/2023 |
| **Mô tả ngắn gọn:** Kiểm tra chức năng import dữ liệu chấm công bằng file excel  của phương thức ImportOfficerAttendace sai dữ liệu |  |

|  |
| --- |
| **Tiền điều kiện:**  Hệ thống đang hiển thị giao diện chọn file để import.  File có định dạng đúng và dữ liệu không hợp lệ. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bước | Hành động | Phản hồi mong muốn từ hệ thống | Pass/Fail | Comment |
| 1 | Người dùng click nút chọn file import | Hệ thống hiển thị giao diện chọn file | Pass |  |
| 2 | Người dùng chọn file | Hệ thống hiển thị dữ liệu lên bảng | Pass |  |
| 3 | Người dùng chọn nút Nhập dữ liệu | Hệ thống hiển thị Confirm box để xác nhận lần nữa | Pass |  |
| 4 | Người dùng ấn nút Đồng ý | Hệ thống thông báo import thất bại | Pass |  |

|  |
| --- |
| **Trạng thái của hệ thống sau khi chạy hàm:**  Hệ thống trở lại trạng thái ban đầu. |

### **Thiết kế test case cho phương thức createSession**:

**Tên phương thức**: createSession

**Kiểu trả về**: Session

**Danh sách tham số**: tham số time định dạng String

**Chi tiết**: Hàm nhận vào 1 chuỗi định dạng thời gian. Sau đó trả về 1 Session. Chuỗi phải có định dạng dd/MM/yyyy HH:mm:ss

Trường dữ liệu của Session:

|  |  |  |
| --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Comment |
| date | Date | Thời gian chấm công |
| isMorningSession | boolean | Có phải ca sáng |
| isAfternoonSession | boolean | Có phải ca chiều |
| hoursLate | double | Số giờ đi muộn |

#### Kỹ thuật sử dụng: kiểm thử hộp trắng và bảng quyết định.

Kiểm thử hộp trắng (độ đo c1): bao hàm tất cả các trường hợp xảy ra trong đoạn code.

Sử dụng Control Flow test:

A diagram of a flowchart

Description automatically generated

Tên class kiểm thử: CreateSessionTest

Các test case:

+ CreateSessionTest\_Success: Kiểm thử trường hợp tạo thành công

+ CreateSessionTest\_FailInvalidString: Kiểm thử trường hợp chuỗi không đúng định dạng

+ CreateSessionTest\_TestMorningSession: Kiểm thử trường hợp thời gian là ca sáng

+ CreateSessionTest\_TestAfternoonSession: Kiểm thử trường hợp thời gian là ca chiều

+ CreateSessionTest\_TestLate: Kiểm thử trường hợp đi muộn

### Thiết kế test case cho usecase: Nhập dữ liệu bằng file excel

#### Test case ImportDataExcel\_Success

|  |  |
| --- | --- |
| **Test case #:** 4.4 | **Tên testcase:** ImportDataExcel\_Success |
| **Hệ thống:** Hệ thống quản lý chấm công | **Hệ thống con:** Hệ thống quản lý nhân sự |
| **Thiết kế bởi:** Nguyễn Thanh Lâm 20200336 | **Ngày thiết kế:** 20/12/2023 |
| **Thực thi bởi:** Hệ thống quản lý chấm công | **Ngày thực thi:** 24/12/2023 |
| **Mô tả ngắn gọn:** Kiểm tra chức năng import dữ liệu |  |

|  |
| --- |
| **Tiền điều kiện:**  Hệ thống đang hiển thị giao diện chọn file để import  File có định dạng và dữ liệu hợp lệ |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bước | Hành động | Phản hồi mong muốn từ hệ thống | Pass/Fail | Comment |
| 1 | Người dùng click nút chọn file import | Hệ thống hiển thị giao diện chọn file | Pass |  |
| 2 | Người dùng chọn file | Hệ thống hiển thị giá trị các cột của file lên 1 bảng. | Pass |  |
| 3 | Người dùng chọn nút Nhập dữ liệu | Hệ thống hiển thị Confirm box để xác nhận lần nữa | Pass |  |
| 4 | Người dùng ấn nút Đồng ý | Hệ thống thông báo import thành công | Pass |  |

|  |
| --- |
| **Trạng thái của hệ thống sau khi chạy hàm:**  Hệ thống nhập dữ liệu về lịch sử import và các thông tin về ca chấm công từ file vào trong database. |

#### Test case ImportDataExcel\_InvalidFileExtension

|  |  |
| --- | --- |
| **Test case #:** 4.5 | **Tên testcase**: ImportDataExcel InvalidFileExtension |
| **Hệ thống:** Hệ thống quản lý chấm công | **Hệ thống con:** Hệ thống quản lý nhân sự |
| **Thiết kế bởi:** Nguyễn Thanh Lâm 20200336 | **Ngày thiết kế:** 20/12/2023 |
| **Thực thi bởi:** Hệ thống quản lý chấm công | **Ngày thực thi:** 24/12/2023 |
| **Mô tả ngắn gọn:** Kiểm tra chức năng import dữ liệu  khi file có định dạng không phải xlsx/csv |  |

|  |
| --- |
| **Tiền điều kiện:**  Hệ thống đang hiển thị giao diện chọn file để import.  File có định dạng không đúng. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bước | Hành động | Phản hồi mong muốn từ hệ thống | Pass/Fail | Comment |
| 1 | Người dùng click nút chọn file import | Hệ thống hiển thị giao diện chọn file | Pass |  |
| 2 | Người dùng chọn file | Hệ thống hiển thị thông báo định dạng file không hợp lệ | Pass |  |

|  |
| --- |
| **Trạng thái của hệ thống sau khi chạy hàm:**  Hệ thống trở lại trạng thái ban đầu. |

#### Test case ImportDataExcel\_InvalidData

|  |  |
| --- | --- |
| **Test case #:** 4.6 | **Tên test case**: ImportDataExcelInvalidData |
| **Hệ thống:** Hệ thống quản lý chấm công | **Hệ thống con:** Hệ thống quản lý nhân sự |
| **Thiết kế bởi:** Nguyễn Thanh Lâm 20200336 | **Ngày thiết kế:** 20/12/2023 |
| **Thực thi bởi:** ImportService | **Ngày thực thi:** 24/12/2023 |
| **Mô tả ngắn gọn:** Kiểm tra chức năng import dữ liệu chấm công bằng file excel khi dữ liệu không hợp lệ |  |

|  |
| --- |
| **Tiền điều kiện:**  Hệ thống đang hiển thị giao diện chọn file để import.  File có định dạng đúng và dữ liệu không hợp lệ. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bước | Hành động | Phản hồi mong muốn từ hệ thống | Pass/Fail | Comment |
| 1 | Người dùng click nút chọn file import | Hệ thống hiển thị giao diện chọn file | Pass |  |
| 2 | Người dùng chọn file | Hệ thống hiển thị dữ liệu lên bảng | Pass |  |
| 3 | Người dùng chọn nút Nhập dữ liệu | Hệ thống hiển thị Confirm box để xác nhận lần nữa | Pass |  |
| 4 | Người dùng ấn nút Đồng ý | Hệ thống thông báo import thất bại | Pass |  |

|  |
| --- |
| **Trạng thái của hệ thống sau khi chạy hàm:**  Hệ thống trở lại trạng thái ban đầu. |