Tài liệu Chức năng Cốt lõi HRM InShield

TechShield Development Team

2025-10-25

# Giới thiệu

## Tổng quan hệ thống HRM InShield

HRM InShield là hệ thống quản lý nhân sự toàn diện được phát triển bằng Spring Boot, cung cấp các chức năng quản lý nhân viên, chấm công, đối tác và tài sản. Hệ thống được thiết kế với kiến trúc MVC hiện đại, hỗ trợ phân quyền người dùng và giao diện web thân thiện.

## Mục đích tài liệu

Tài liệu này mô tả chi tiết 4 chức năng cốt lõi của hệ thống HRM InShield:

* Quản lý Nhân viên (Employee Management)
* Chấm công (Attendance Management)
* Quản lý Đối tác (Partner Management)
* Quản lý Tài sản (Equipment Management)

Mỗi chức năng bao gồm: khảo sát hệ thống, kịch bản sử dụng, biểu đồ thực thể, thiết kế cơ sở dữ liệu và giao diện người dùng.

## Phạm vi tài liệu

Tài liệu tập trung vào:

* Kiến trúc và công nghệ sử dụng
* Luồng xử lý nghiệp vụ (Business Process)
* Thiết kế cơ sở dữ liệu
* Giao diện người dùng
* Biểu đồ sequence và ERD

# Chức năng Quản lý Nhân viên

## Khảo sát hệ thống

### Kiến trúc hệ thống

Hệ thống HRM InShield được xây dựng trên nền tảng Spring Boot với kiến trúc MVC:

* **Backend**: Spring Boot 3.x, Spring Security, Spring Data JPA
* **Frontend**: Thymeleaf templates, Bootstrap CSS, JavaScript
* **Database**: MariaDB với Liquibase migration
* **Authentication**: Spring Security với role-based access control

### Công nghệ sử dụng

| Công nghệ | Mô tả | Phiên bản |
| --- | --- | --- |
| Spring Boot | Framework chính cho backend | 3.x |
| Spring Security | Xác thực và phân quyền | 6.x |
| Spring Data JPA | ORM và truy vấn database | 3.x |
| Thymeleaf | Template engine cho frontend | 3.x |
| MariaDB | Hệ quản trị cơ sở dữ liệu | 10.x |
| Liquibase | Database migration tool | 4.x |
| Bootstrap | CSS framework cho UI | 5.x |

## Kịch bản sử dụng

### Use Case 1: Xem danh sách nhân viên

**Mô tả**: HR/Admin xem danh sách tất cả nhân viên trong hệ thống

**Luồng thực hiện**: 1. Người dùng đăng nhập vào hệ thống 2. Truy cập menu "Nhân viên" 3. Hệ thống hiển thị danh sách nhân viên với phân trang 4. Người dùng có thể tìm kiếm và lọc theo điều kiện

**Kết quả**: Danh sách nhân viên được hiển thị với thông tin cơ bản

### Use Case 2: Thêm nhân viên mới

**Mô tả**: HR/Admin thêm nhân viên mới vào hệ thống

**Luồng thực hiện**: 1. Click nút "Thêm nhân viên" 2. Điền form thông tin nhân viên: - Thông tin cơ bản: Mã, Họ tên, Email, Điện thoại - Thông tin công việc: Phòng ban, Chức vụ, Ngày vào làm, Lương - Thông tin quản lý: Quản lý trực tiếp, Tài khoản liên kết - Phân quyền: Chọn roles (ADMIN, HR, MANAGER, EMPLOYEE) 3. Click "Lưu" 4. Hệ thống tạo tài khoản đăng nhập (nếu có email) 5. Gửi email thông tin tài khoản cho nhân viên

**Kết quả**: Nhân viên mới được tạo và có tài khoản đăng nhập

### Use Case 3: Cập nhật thông tin nhân viên

**Mô tả**: HR/Admin chỉnh sửa thông tin nhân viên hiện có

**Luồng thực hiện**: 1. Click "Chỉnh sửa" trên nhân viên cần sửa 2. Cập nhật thông tin trong form 3. Click "Lưu" 4. Hệ thống cập nhật thông tin

**Kết quả**: Thông tin nhân viên được cập nhật

### Use Case 4: Xóa nhân viên

**Mô tả**: HR/Admin xóa nhân viên khỏi hệ thống

**Luồng thực hiện**: 1. Click "Xóa" trên nhân viên cần xóa 2. Xác nhận xóa 3. Hệ thống xóa nhân viên và tài khoản liên kết

**Kết quả**: Nhân viên bị xóa khỏi hệ thống

## Biểu đồ Sequence

### Luồng xem danh sách nhân viên

Failed to generate image: undefined method `to\_sym' for nil:NilClass  
@startuml  
actor "Người dùng (HR/Admin)" as User  
participant "Giao diện Quản lý Nhân viên" as View  
participant "EmployeeController" as Controller  
participant "EmployeeService" as Service  
participant "CSDL (EmployeeRepository)" as DB  
  
== Xem danh sách nhân viên ==  
User -> View: Mở trang Quản lý Nhân viên  
View -> Controller: GET /cms/employees  
Controller -> Service: Lấy danh sách nhân viên (phân trang)  
Service -> DB: Truy vấn dữ liệu nhân viên  
DB --> Service: Danh sách nhân viên  
Service --> Controller: Trả danh sách nhân viên  
Controller -> View: Render giao diện danh sách  
User <- View: Hiển thị danh sách nhân viên  
  
== Thêm nhân viên ==  
User -> View: Nhập thông tin nhân viên mới  
View -> Controller: POST /cms/employees (EmployeeRequest)  
Controller -> Service: Gọi hàm create(request)  
Service -> Service: Kiểm tra và ánh xạ dữ liệu (validate + mapping)  
Service -> DB: Lưu nhân viên mới vào CSDL  
DB --> Service: Trả về đối tượng nhân viên đã lưu  
Service --> Controller: Trả thông báo thành công  
Controller -> View: Reload danh sách + hiển thị thông báo “Thêm mới thành công”  
User <- View: Xem danh sách có nhân viên mới  
  
== Cập nhật nhân viên ==  
User -> View: Chọn chỉnh sửa thông tin nhân viên  
View -> Controller: POST /cms/employees/{id}/update (EmployeeRequest)  
Controller -> Service: Gọi hàm update(id, request)  
Service -> DB: Tìm nhân viên theo ID  
DB --> Service: Trả về thông tin nhân viên  
Service -> Service: Cập nhật thông tin mới vào entity  
Service -> DB: Ghi lại thông tin đã cập nhật  
DB --> Service: Xác nhận thành công  
Service --> Controller: Thông báo cập nhật thành công  
Controller -> View: Cập nhật lại danh sách  
User <- View: Xem danh sách đã thay đổi  
  
== Xóa nhân viên ==  
User -> View: Chọn xóa nhân viên  
View -> Controller: POST /cms/employees/{id}/delete  
Controller -> Service: Gọi hàm delete(id)  
Service -> DB: Xóa bản ghi nhân viên  
DB --> Service: Xác nhận xóa thành công  
Service --> Controller: Thông báo xóa thành công  
Controller -> View: Cập nhật lại danh sách nhân viên  
User <- View: Danh sách sau khi xóa  
@enduml

### Luồng thêm nhân viên với tài khoản

Failed to generate image: undefined method `to\_sym' for nil:NilClass  
@startuml  
actor "Người dùng (HR/Admin)" as User  
participant "Giao diện Quản lý Nhân viên" as View  
participant "EmployeeController" as Controller  
participant "EmployeeService" as Service  
participant "UserAccountRepository" as UserRepo  
participant "EmployeeRepository" as EmpRepo  
participant "RoleRepository" as RoleRepo  
participant "MailService" as Mail  
  
== Thêm nhân viên ==  
User -> View: Nhập thông tin nhân viên mới (EmployeeRequest)  
View -> Controller: POST /cms/employees (EmployeeRequest)  
  
Controller -> Service: create(request)  
  
activate Service  
Service -> EmpRepo: Lưu nhân viên mới (employee)  
EmpRepo --> Service: Nhân viên đã lưu (có ID)  
  
alt Có email trong request  
 Service -> Service: generateUsername(employee)\ngenerateTempPassword()  
 Service -> RoleRepo: Lấy role mặc định EMPLOYEE  
 RoleRepo --> Service: Trả về Role EMPLOYEE  
  
 Service -> UserRepo: Lưu tài khoản UserAccount\n(username, password mã hóa, email, roles)  
 UserRepo --> Service: UserAccount đã lưu  
  
 Service -> EmpRepo: Gán userAccount vào employee và cập nhật  
 EmpRepo --> Service: Employee đã cập nhật  
  
 Service -> Mail: Gửi email thông tin tài khoản\n(Username, mật khẩu tạm thời)  
 Mail --> Service: Xác nhận gửi thành công  
end  
  
Service --> Controller: Trả về EmployeeResponse (kết quả tạo thành công)  
deactivate Service  
  
Controller -> View: Hiển thị thông báo “Thêm nhân viên thành công”  
User <- View: Xem danh sách có nhân viên mới  
@enduml

## Biểu đồ Thực thể (ERD)

Failed to generate image: undefined method `to\_sym' for nil:NilClass  
@startuml  
!define TABLE(name,desc) class name as "desc" << (T,#FFAAAA) >>  
!define PK(x) <b><color:#b8861b><&key></color> x</b>  
!define FK(x) <color:#aaaaaa></color> x  
!define NN(x) <color:#ef4444></color> x  
!define UQ(x) <color:#3b82f6></color> x  
  
package "Employee Management" {  
 TABLE(employees, "employees") {  
 PK(id) BIGINT AUTO\_INCREMENT  
 UQ(code) VARCHAR(50) NOT NULL  
 NN(full\_name) VARCHAR(150) NOT NULL  
 UQ(email) VARCHAR(150) NOT NULL  
 phone VARCHAR(30)  
 department VARCHAR(100)  
 position VARCHAR(100)  
 NN(hire\_date) DATE NOT NULL  
 termination\_date DATE  
 employment\_status VARCHAR(50)  
 base\_salary DECIMAL(12,2)  
 FK(manager\_id) BIGINT  
 FK(user\_id) BIGINT  
 }  
  
 TABLE(users, "users") {  
 PK(id) BIGINT AUTO\_INCREMENT  
 UQ(username) VARCHAR(100) NOT NULL  
 UQ(email) VARCHAR(150) NOT NULL  
 NN(password) VARCHAR(255) NOT NULL  
 NN(enabled) BOOLEAN DEFAULT true  
 NN(account\_non\_locked) BOOLEAN DEFAULT true  
 last\_login\_at TIMESTAMP  
 }  
  
 TABLE(roles, "roles") {  
 PK(id) BIGINT AUTO\_INCREMENT  
 UQ(name) VARCHAR(50) NOT NULL  
 }  
  
 TABLE(user\_roles, "user\_roles") {  
 PK(user\_id) BIGINT NOT NULL  
 PK(role\_id) BIGINT NOT NULL  
 }  
}  
  
' Relationships  
employees ||--o{ employees : "manager\_id"  
employees ||--o| users : "user\_id"  
users ||--o{ user\_roles : "user\_id"  
roles ||--o{ user\_roles : "role\_id"  
  
note right of employees  
 \*\*Employee Entity\*\*  
 - Quản lý thông tin nhân viên  
 - Liên kết với tài khoản người dùng  
 - Hỗ trợ cấu trúc phân cấp quản lý  
end note  
  
note right of users  
 \*\*User Account\*\*  
 - Tài khoản đăng nhập hệ thống  
 - Hỗ trợ xác thực và phân quyền  
 - Liên kết với thông tin nhân viên  
end note  
  
note right of roles  
 \*\*Role Management\*\*  
 - Định nghĩa các vai trò trong hệ thống  
 - Hỗ trợ phân quyền chi tiết  
 - Liên kết nhiều-nhiều với users  
end note  
@enduml

## Thiết kế Cơ sở Dữ liệu

### Bảng employees

| Cột | Kiểu dữ liệu | Ràng buộc | Mô tả | id |
| --- | --- | --- | --- | --- |
| BIGINT AUTO\_INCREMENT | PK | Khóa chính | code | VARCHAR(50) |
| NOT NULL, UNIQUE | Mã nhân viên (EMP-001) | full\_name | VARCHAR(150) | NOT NULL |
| Họ và tên đầy đủ | email | VARCHAR(150) | NOT NULL, UNIQUE | Email liên hệ |
| phone | VARCHAR(30) | NULL | Số điện thoại | department |
| VARCHAR(100) | NULL | Phòng ban | position | VARCHAR(100) |
| NULL | Chức vụ | hire\_date | DATE | NOT NULL |
| Ngày vào làm | termination\_date | DATE | NULL | Ngày nghỉ việc |
| employment\_status | VARCHAR(50) | NULL | Trạng thái làm việc | base\_salary |
| DECIMAL(12,2) | NULL | Lương cơ bản | manager\_id | BIGINT |
| FK | ID quản lý trực tiếp | user\_id | BIGINT | FK |

### Bảng users

| Cột | Kiểu dữ liệu | Ràng buộc | Mô tả | id |
| --- | --- | --- | --- | --- |
| BIGINT AUTO\_INCREMENT | PK | Khóa chính | username | VARCHAR(100) |
| NOT NULL, UNIQUE | Tên đăng nhập | email | VARCHAR(150) | NOT NULL, UNIQUE |
| Email đăng nhập | password | VARCHAR(255) | NOT NULL | Mật khẩu (mã hóa) |
| enabled | BOOLEAN | NOT NULL, DEFAULT true | Tài khoản có hoạt động | account\_non\_locked |
| BOOLEAN | NOT NULL, DEFAULT true | Tài khoản có bị khóa | last\_login\_at | TIMESTAMP |

### Bảng roles

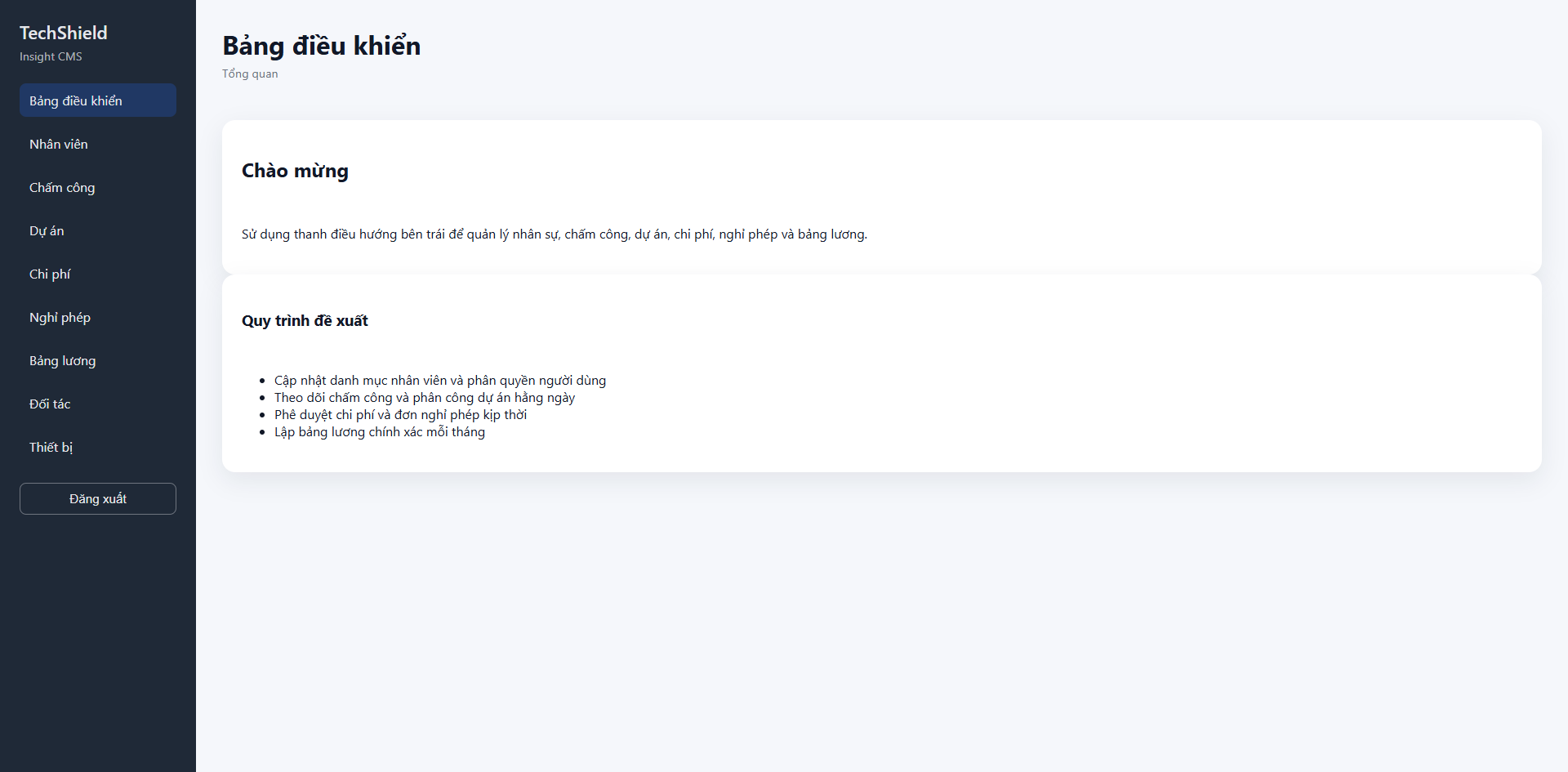
| Cột | Kiểu dữ liệu | Ràng buộc | Mô tả | id |
| --- | --- | --- | --- | --- |
| BIGINT AUTO\_INCREMENT | PK | Khóa chính | name | VARCHAR(50) |

### Bảng user\_roles

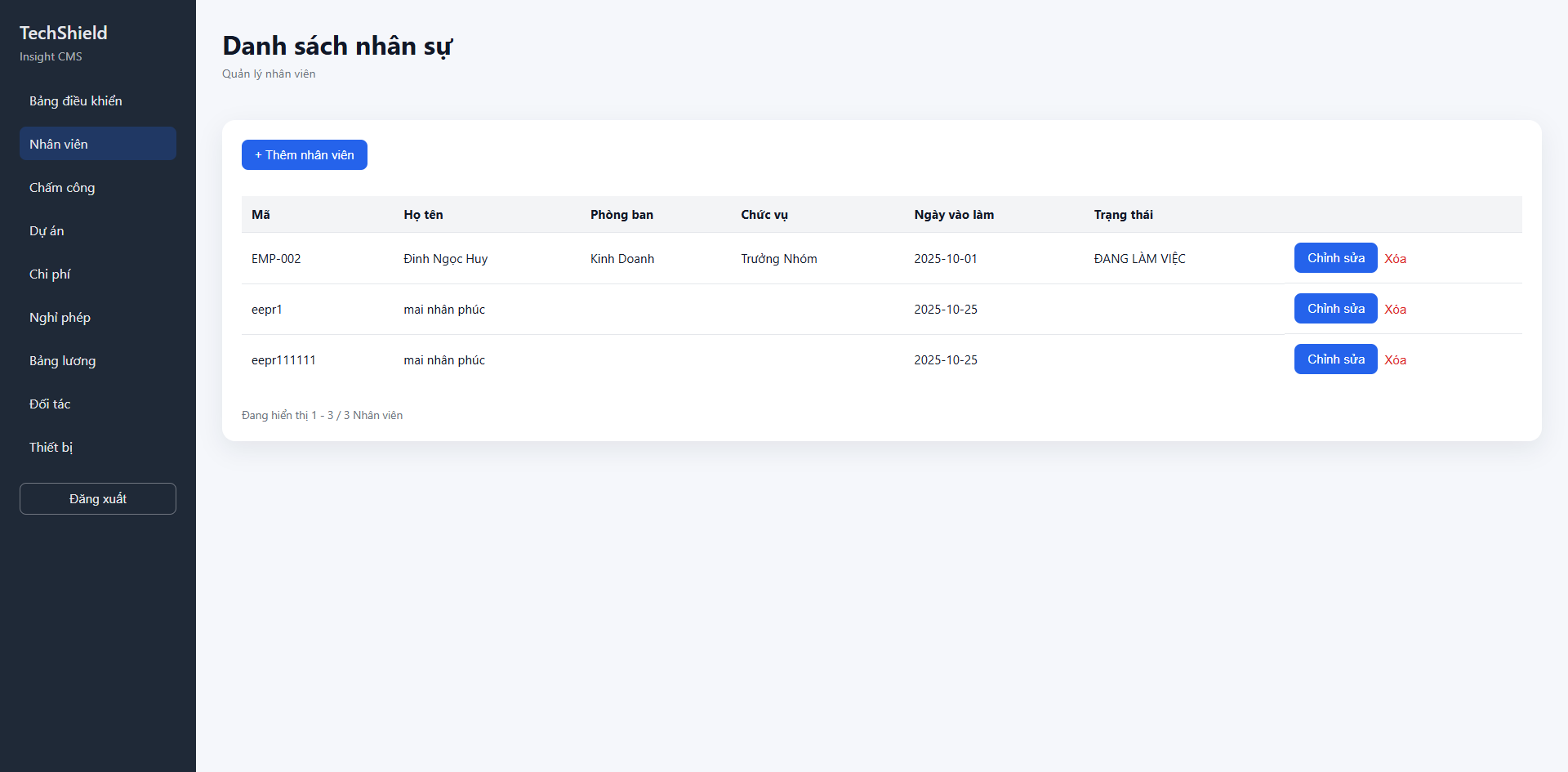
| Cột | Kiểu dữ liệu | Ràng buộc | Mô tả | user\_id |
| --- | --- | --- | --- | --- |
| BIGINT | PK, FK | ID người dùng | role\_id | BIGINT |

## Giao diện Người dùng

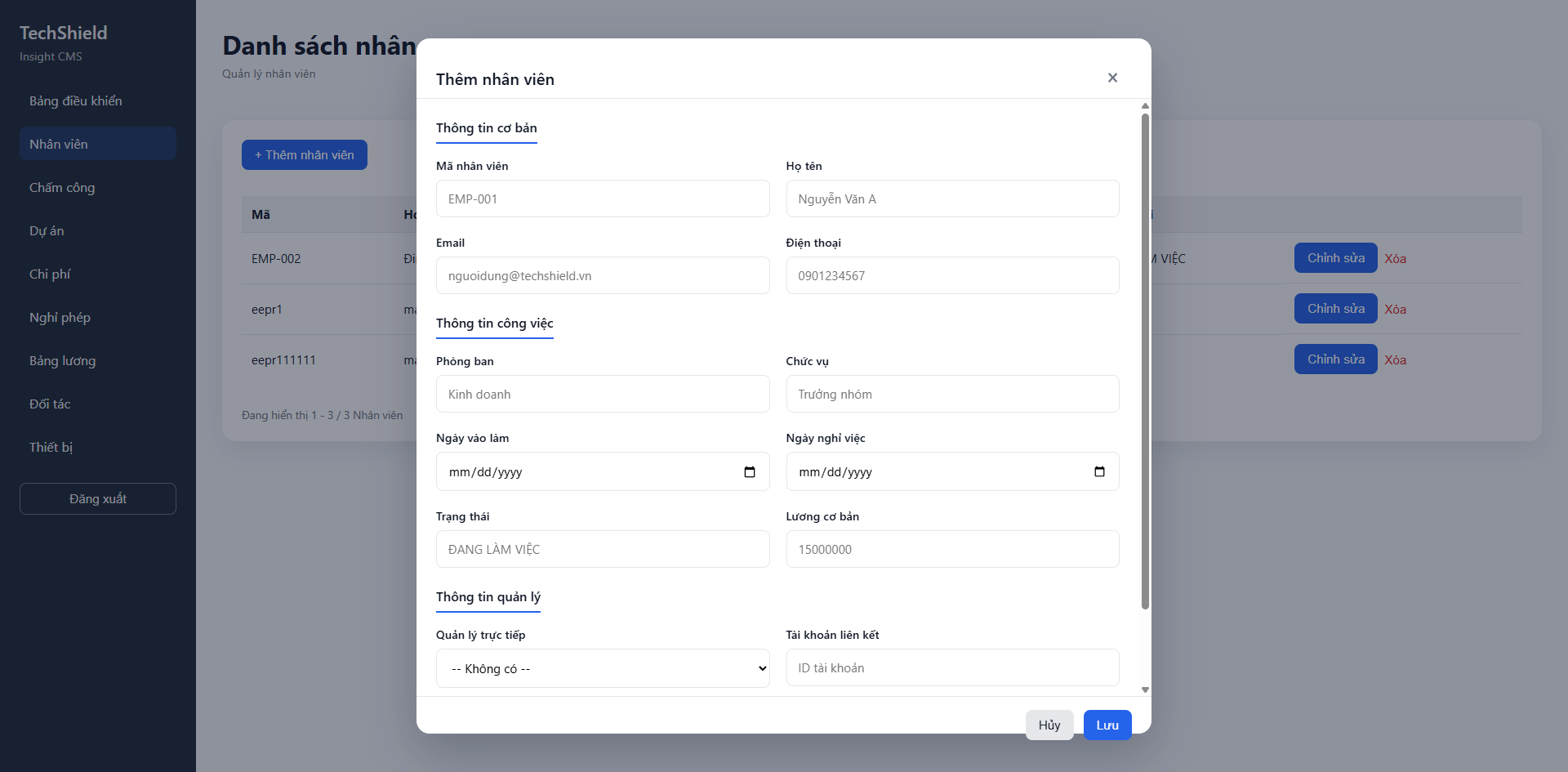
### Màn hình Dashboard



### Màn hình Danh sách Nhân viên



### Màn hình Form Thêm Nhân viên



# Chức năng Chấm công

## Khảo sát hệ thống

Chức năng chấm công cho phép quản lý thời gian làm việc của nhân viên, bao gồm:

* Ghi nhận giờ vào/ra
* Tính toán số giờ làm việc tự động
* Theo dõi lịch sử chấm công
* Báo cáo thời gian làm việc

## Kịch bản sử dụng

### Use Case 1: Ghi nhận chấm công

**Mô tả**: HR/Admin ghi nhận chấm công cho nhân viên

**Luồng thực hiện**: 1. Truy cập menu "Chấm công" 2. Click "Thêm chấm công" 3. Chọn nhân viên từ dropdown 4. Nhập ngày làm việc 5. Nhập giờ vào và giờ ra 6. Hệ thống tự động tính số giờ làm việc 7. Thêm ghi chú (tùy chọn) 8. Click "Lưu"

**Kết quả**: Bản ghi chấm công được tạo

### Use Case 2: Xem lịch sử chấm công

**Mô tả**: HR/Admin xem lịch sử chấm công của nhân viên

**Luồng thực hiện**: 1. Truy cập danh sách chấm công 2. Xem thông tin: Nhân viên, Ngày, Giờ vào/ra, Số giờ 3. Có thể lọc theo nhân viên hoặc khoảng thời gian

**Kết quả**: Hiển thị lịch sử chấm công

### Use Case 3: Chỉnh sửa chấm công

**Mô tả**: HR/Admin chỉnh sửa thông tin chấm công

**Luồng thực hiện**: 1. Click "Chỉnh sửa" trên bản ghi cần sửa 2. Cập nhật thông tin 3. Click "Lưu"

**Kết quả**: Thông tin chấm công được cập nhật

## Biểu đồ Sequence

Failed to generate image: undefined method `to\_sym' for nil:NilClass  
@startuml  
actor "Người dùng (HR/Admin)" as User  
participant "Giao diện Quản lý Chấm công" as View  
participant "AttendanceController" as Controller  
participant "AttendanceService" as Service  
participant "CSDL (AttendanceRepository)" as DB  
  
== Xem danh sách ==  
User -> View: Mở trang /cms/attendance  
View -> Controller: GET /cms/attendance  
Controller -> Service: Lấy danh sách chấm công (phân trang)  
Service -> DB: Truy vấn dữ liệu từ bảng attendance\_record  
DB --> Service: Trả kết quả danh sách  
Service --> Controller: Danh sách AttendanceResponse  
Controller -> View: Render HTML hiển thị danh sách  
  
== Thêm chấm công ==  
User -> View: Nhập form + nhấn “Lưu”  
View -> Controller: POST /cms/attendance  
Controller -> Service: Gọi create(request)  
Service -> DB: Lưu bản ghi mới vào bảng attendance\_record  
DB --> Service: Trả kết quả thành công  
Service --> Controller: Trả thông báo thành công  
Controller -> View: Reload danh sách + hiển thị thông báo  
  
== Cập nhật ==  
User -> View: Sửa thông tin bản ghi  
View -> Controller: POST /cms/attendance/{id}/update  
Controller -> Service: Gọi update(id, request)  
Service -> DB: Tìm và cập nhật bản ghi  
DB --> Service: Xác nhận cập nhật  
Service --> Controller: Trả thông báo thành công  
Controller -> View: Reload danh sách  
  
== Xóa ==  
User -> View: Bấm “Xóa”  
View -> Controller: POST /cms/attendance/{id}/delete  
Controller -> Service: Gọi delete(id)  
Service -> DB: Xóa bản ghi  
DB --> Service: Xác nhận xóa  
Service --> Controller: Thông báo thành công  
Controller -> View: Reload danh sách sau khi xóa  
@enduml

## Biểu đồ Thực thể (ERD)

Failed to generate image: undefined method `to\_sym' for nil:NilClass  
@startuml  
!define TABLE(name,desc) class name as "desc" << (T,#FFAAAA) >>  
!define PK(x) <b><color:#b8861b><&key></color> x</b>  
!define FK(x) <color:#aaaaaa></color> x  
!define NN(x) <color:#ef4444></color> x  
!define UQ(x) <color:#3b82f6></color> x  
  
package "Attendance Management" {  
 TABLE(attendance\_records, "attendance\_records") {  
 PK(id) BIGINT AUTO\_INCREMENT  
 FK(employee\_id) BIGINT NOT NULL  
 NN(work\_date) DATE NOT NULL  
 check\_in TIME  
 check\_out TIME  
 worked\_hours DECIMAL(5,2)  
 notes VARCHAR(255)  
 }  
  
 TABLE(employees, "employees") {  
 PK(id) BIGINT AUTO\_INCREMENT  
 UQ(code) VARCHAR(50) NOT NULL  
 NN(full\_name) VARCHAR(150) NOT NULL  
 UQ(email) VARCHAR(150) NOT NULL  
 phone VARCHAR(30)  
 department VARCHAR(100)  
 position VARCHAR(100)  
 NN(hire\_date) DATE NOT NULL  
 termination\_date DATE  
 employment\_status VARCHAR(50)  
 base\_salary DECIMAL(12,2)  
 FK(manager\_id) BIGINT  
 FK(user\_id) BIGINT  
 }  
}  
  
' Relationships  
attendance\_records ||--|| employees : "employee\_id"  
  
note right of attendance\_records  
 \*\*Attendance Record\*\*  
 - Ghi nhận chấm công hàng ngày  
 - Tự động tính số giờ làm việc  
 - Hỗ trợ ghi chú và trạng thái  
 - Unique constraint: (employee\_id, work\_date)  
end note  
  
note right of employees  
 \*\*Employee Reference\*\*  
 - Thông tin nhân viên để chấm công  
 - Liên kết 1-nhiều với attendance\_records  
 - Hỗ trợ báo cáo theo nhân viên  
end note  
@enduml

## Thiết kế Cơ sở Dữ liệu

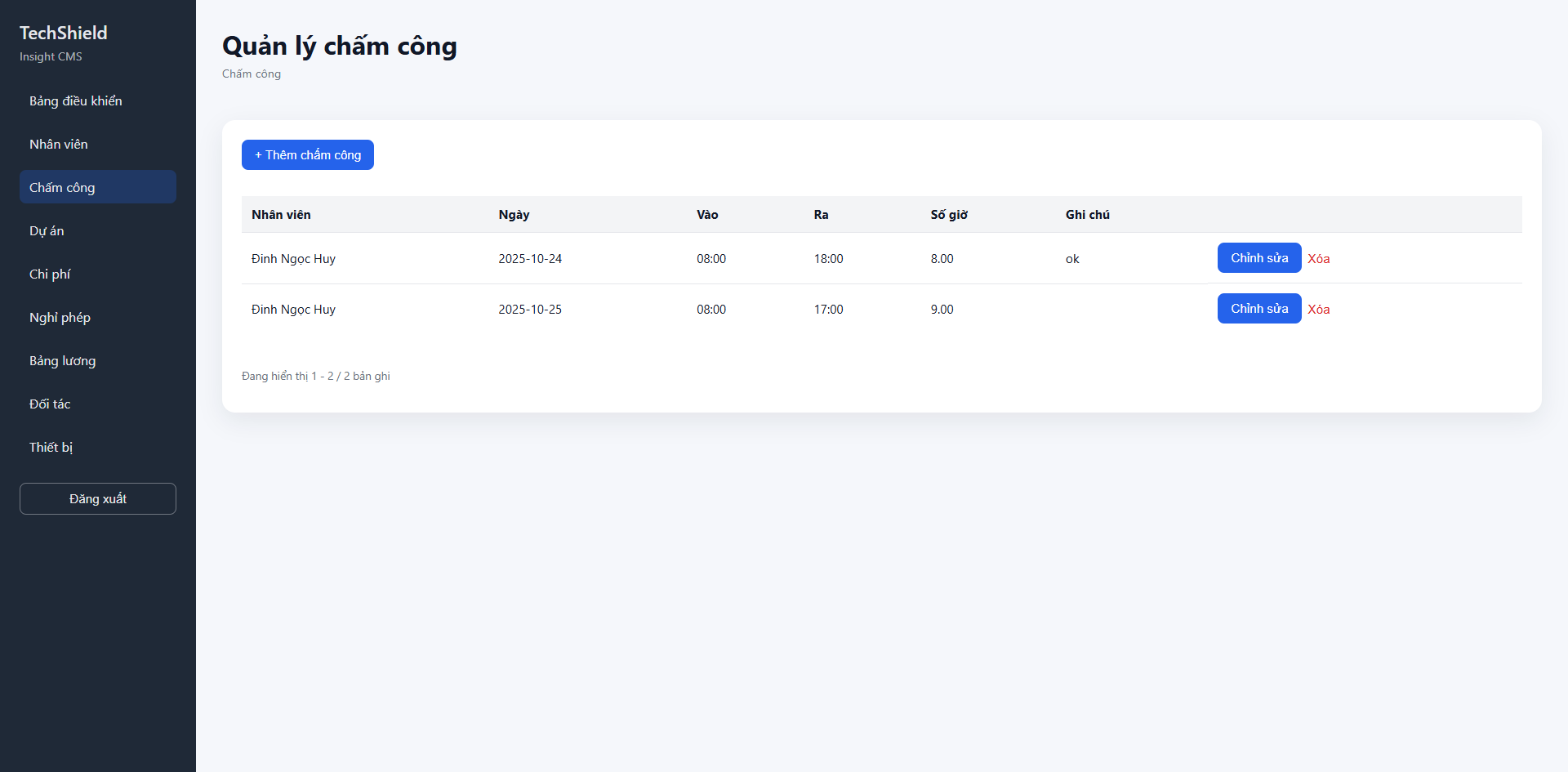
### Bảng attendance\_records

| Cột | Kiểu dữ liệu | Ràng buộc | Mô tả | id |
| --- | --- | --- | --- | --- |
| BIGINT AUTO\_INCREMENT | PK | Khóa chính | employee\_id | BIGINT |
| NOT NULL, FK | ID nhân viên | work\_date | DATE | NOT NULL |
| Ngày làm việc | check\_in | TIME | NULL | Giờ vào làm |
| check\_out | TIME | NULL | Giờ ra về | worked\_hours |
| DECIMAL(5,2) | NULL | Số giờ làm việc | notes | VARCHAR(255) |

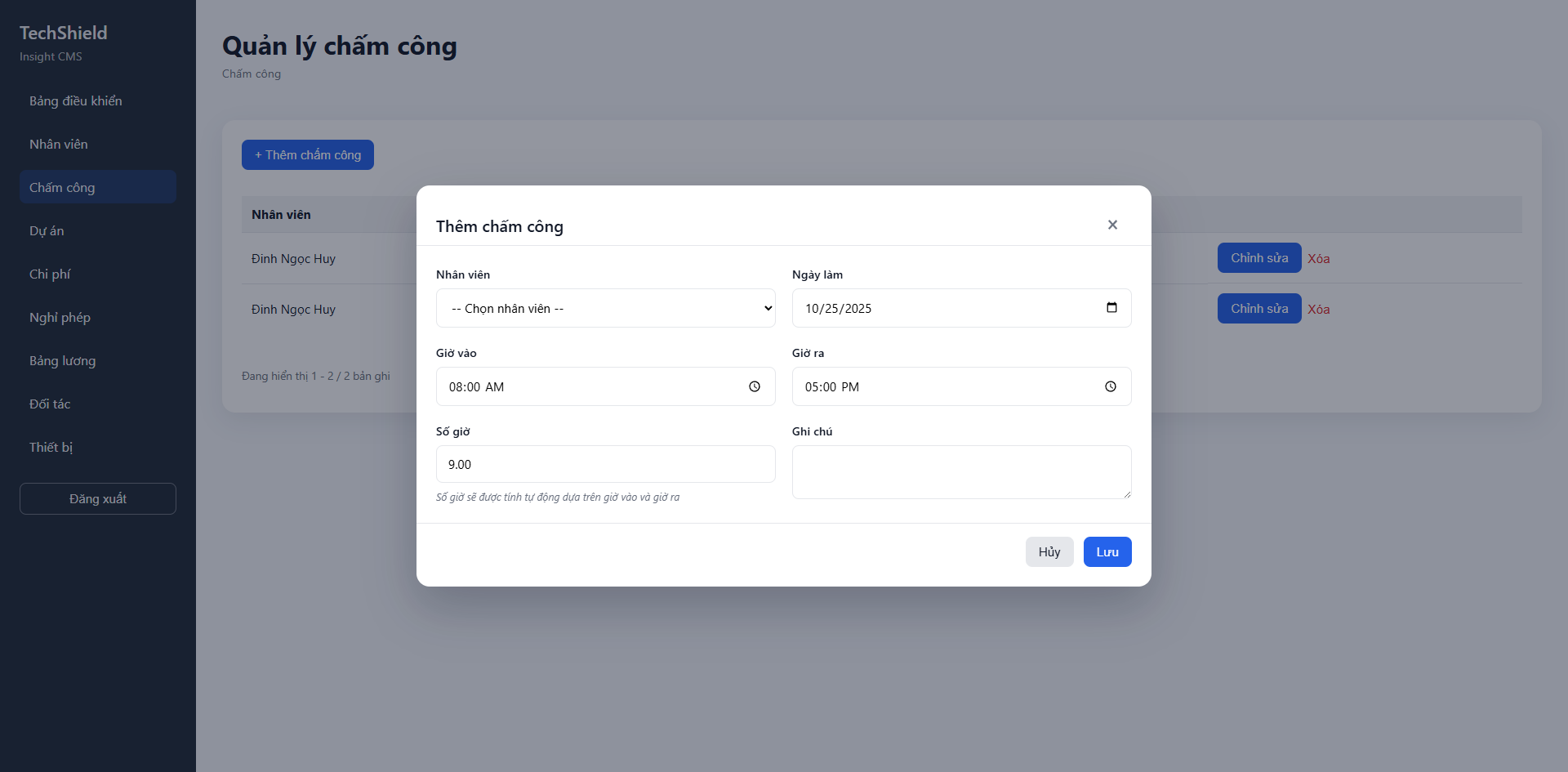
**Ràng buộc đặc biệt**: - Unique constraint: (employee\_id, work\_date) - Mỗi nhân viên chỉ có 1 bản ghi chấm công/ngày

## Giao diện Người dùng

### Màn hình Danh sách Chấm công



### Màn hình Form Thêm Chấm công



# Chức năng Quản lý Đối tác

## Khảo sát hệ thống

Chức năng quản lý đối tác cho phép:

* Quản lý thông tin đối tác (nhà cung cấp, khách hàng, nhà phân phối)
* Phân loại và quản lý trạng thái đối tác
* Tìm kiếm và lọc đối tác
* Theo dõi lịch sử hợp tác

## Kịch bản sử dụng

### Use Case 1: Thêm đối tác mới

**Mô tả**: HR/Admin thêm đối tác mới vào hệ thống

**Luồng thực hiện**: 1. Truy cập menu "Đối tác" 2. Click "Thêm đối tác" 3. Điền thông tin: - Mã đối tác (PART-001) - Tên đối tác - Email, Số điện thoại, Địa chỉ - Loại đối tác (Nhà cung cấp, Khách hàng, Nhà phân phối, Khác) - Trạng thái (Hoạt động, Không hoạt động, Tạm dừng) - Ngày bắt đầu hợp tác - Ghi chú 4. Click "Lưu"

**Kết quả**: Đối tác mới được tạo

### Use Case 2: Quản lý danh sách đối tác

**Mô tả**: HR/Admin xem và quản lý danh sách đối tác

**Luồng thực hiện**: 1. Xem danh sách đối tác 2. Sử dụng bộ lọc: - Tìm theo tên - Lọc theo trạng thái - Lọc theo loại đối tác 3. Chỉnh sửa hoặc xóa đối tác

**Kết quả**: Quản lý hiệu quả danh sách đối tác

## Biểu đồ Sequence

Failed to generate image: undefined method `to\_sym' for nil:NilClass  
@startuml  
actor "Người dùng (HR/Admin)" as User  
participant "Giao diện Quản lý Đối tác" as View  
participant "PartnerController" as Controller  
participant "PartnerService" as Service  
participant "CSDL (PartnerRepository)" as DB  
  
== Xem danh sách đối tác ==  
User -> View: Mở trang Quản lý Đối tác  
View -> Controller: GET /cms/partners  
Controller -> Service: Lấy danh sách đối tác (có lọc)  
Service -> DB: Truy vấn dữ liệu đối tác với filters  
DB --> Service: Danh sách đối tác  
Service --> Controller: Danh sách PartnerResponse  
Controller -> View: Render giao diện danh sách  
User <- View: Hiển thị danh sách đối tác  
  
== Thêm đối tác ==  
User -> View: Nhập thông tin đối tác mới  
View -> Controller: POST /cms/partners (PartnerRequest)  
Controller -> Service: Gọi hàm create(request)  
Service -> Service: Kiểm tra và ánh xạ dữ liệu (validate + mapping)  
Service -> DB: Lưu đối tác mới vào CSDL  
DB --> Service: Trả về đối tượng đối tác đã lưu  
Service --> Controller: Trả thông báo thành công  
Controller -> View: Reload danh sách + hiển thị thông báo "Thêm mới thành công"  
User <- View: Xem danh sách có đối tác mới  
  
== Cập nhật đối tác ==  
User -> View: Chọn chỉnh sửa thông tin đối tác  
View -> Controller: POST /cms/partners/{id}/update (PartnerRequest)  
Controller -> Service: Gọi hàm update(id, request)  
Service -> DB: Tìm đối tác theo ID  
DB --> Service: Trả về thông tin đối tác  
Service -> Service: Cập nhật thông tin mới vào entity  
Service -> DB: Ghi lại thông tin đã cập nhật  
DB --> Service: Xác nhận thành công  
Service --> Controller: Thông báo cập nhật thành công  
Controller -> View: Cập nhật lại danh sách  
User <- View: Xem danh sách đã thay đổi  
  
== Xóa đối tác ==  
User -> View: Chọn xóa đối tác  
View -> Controller: POST /cms/partners/{id}/delete  
Controller -> Service: Gọi hàm delete(id)  
Service -> DB: Xóa bản ghi đối tác  
DB --> Service: Xác nhận xóa thành công  
Service --> Controller: Thông báo xóa thành công  
Controller -> View: Cập nhật lại danh sách đối tác  
User <- View: Danh sách sau khi xóa  
  
== Lọc đối tác ==  
User -> View: Nhập điều kiện lọc (tên, trạng thái, loại)  
View -> Controller: GET /cms/partners?filters  
Controller -> Service: Gọi hàm findByFilters(filters)  
Service -> DB: Truy vấn với điều kiện lọc  
DB --> Service: Kết quả lọc  
Service --> Controller: Danh sách đối tác đã lọc  
Controller -> View: Hiển thị kết quả lọc  
User <- View: Danh sách đối tác theo điều kiện  
@enduml

## Biểu đồ Thực thể (ERD)

Failed to generate image: undefined method `to\_sym' for nil:NilClass  
@startuml  
!define TABLE(name,desc) class name as "desc" << (T,#FFAAAA) >>  
!define PK(x) <b><color:#b8861b><&key></color> x</b>  
!define FK(x) <color:#aaaaaa></color> x  
!define NN(x) <color:#ef4444></color> x  
!define UQ(x) <color:#3b82f6></color> x  
  
package "Partner Management" {  
 TABLE(partners, "partners") {  
 PK(id) BIGINT AUTO\_INCREMENT  
 UQ(code) VARCHAR(50) NOT NULL  
 NN(name) VARCHAR(150) NOT NULL  
 email VARCHAR(150)  
 phone VARCHAR(30)  
 address VARCHAR(255)  
 NN(partner\_type) VARCHAR(50) NOT NULL  
 NN(status) VARCHAR(50) NOT NULL  
 start\_date DATE  
 notes VARCHAR(500)  
 }  
}  
  
note right of partners  
 \*\*Partner Entity\*\*  
 - Quản lý thông tin đối tác  
 - Hỗ trợ phân loại đối tác (Nhà cung cấp, Khách hàng, Nhà phân phối, Khác)  
 - Quản lý trạng thái (Hoạt động, Không hoạt động, Tạm dừng)  
 - Lưu trữ thông tin liên hệ và ghi chú  
end note  
  
note bottom of partners  
 \*\*Partner Types:\*\*  
 - SUPPLIER: Nhà cung cấp  
 - CUSTOMER: Khách hàng  
 - DISTRIBUTOR: Nhà phân phối  
 - OTHER: Khác  
  
 \*\*Status Values:\*\*  
 - ACTIVE: Hoạt động  
 - INACTIVE: Không hoạt động  
 - SUSPENDED: Tạm dừng  
end note  
@enduml

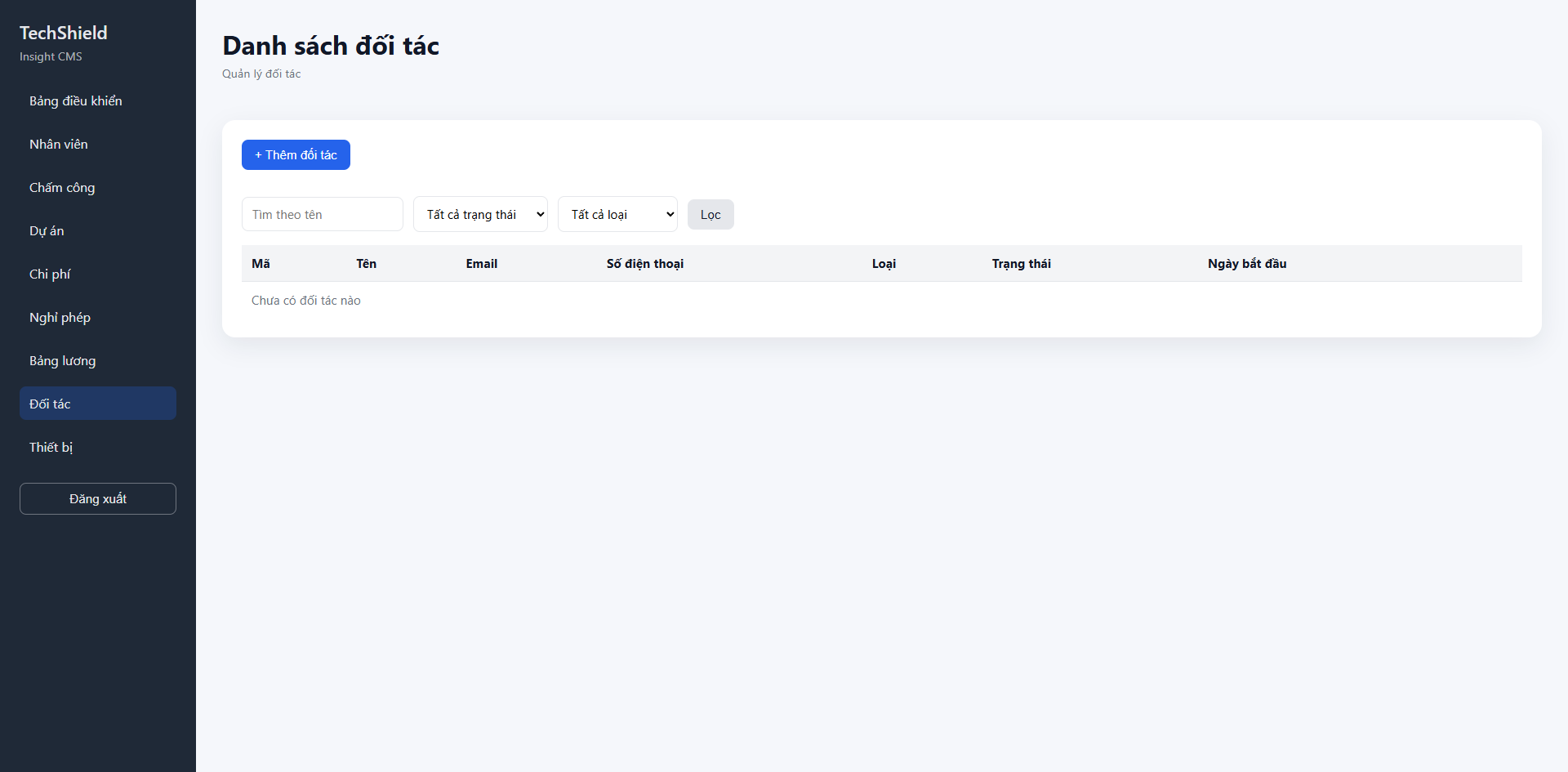
## Thiết kế Cơ sở Dữ liệu

### Bảng partners

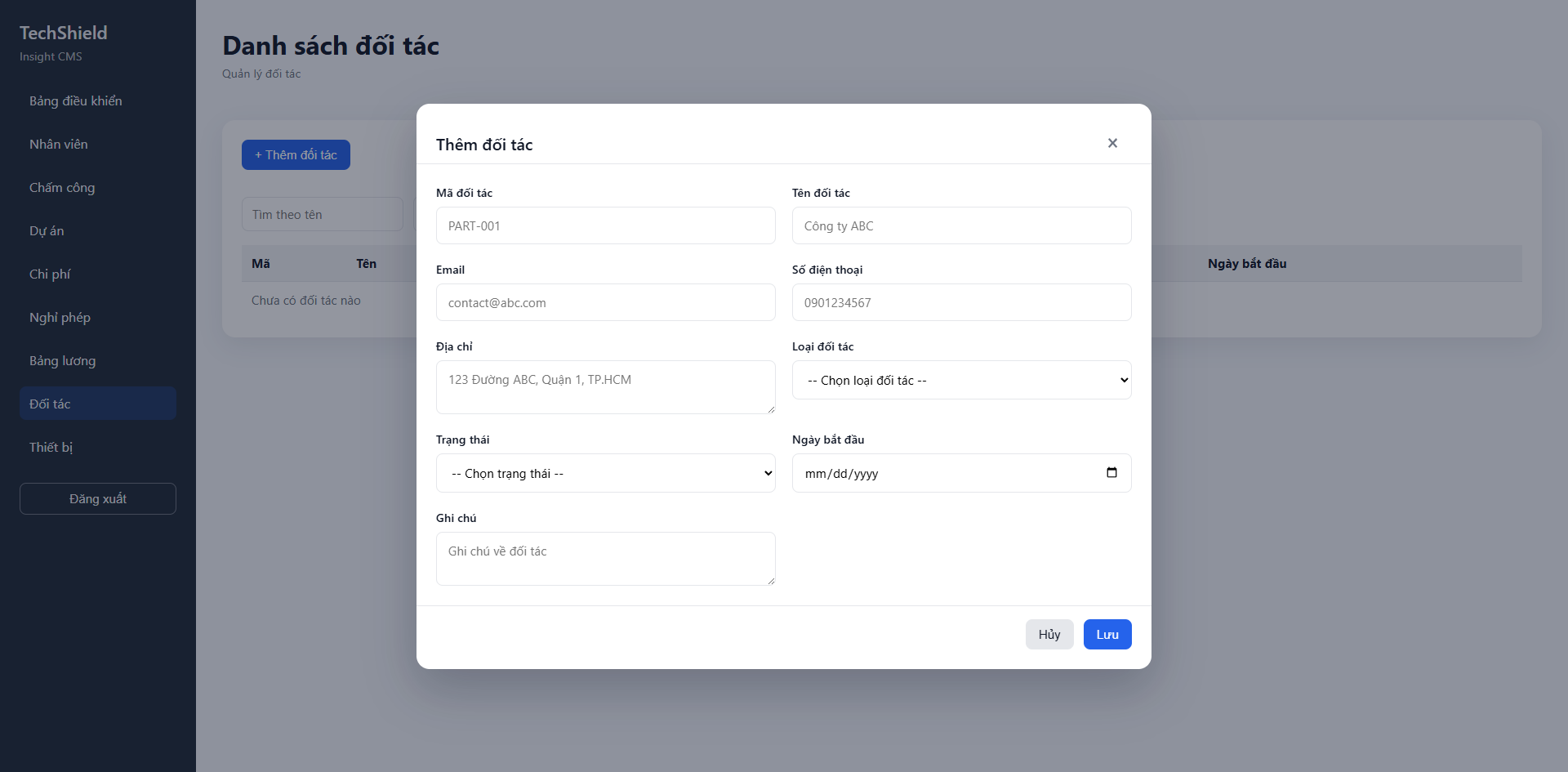
| Cột | Kiểu dữ liệu | Ràng buộc | Mô tả | id |
| --- | --- | --- | --- | --- |
| BIGINT AUTO\_INCREMENT | PK | Khóa chính | code | VARCHAR(50) |
| NOT NULL, UNIQUE | Mã đối tác (PART-001) | name | VARCHAR(150) | NOT NULL |
| Tên đối tác | email | VARCHAR(150) | NULL | Email liên hệ |
| phone | VARCHAR(30) | NULL | Số điện thoại | address |
| VARCHAR(255) | NULL | Địa chỉ | partner\_type | VARCHAR(50) |
| NOT NULL | Loại đối tác (SUPPLIER, CUSTOMER, DISTRIBUTOR, OTHER) | status | VARCHAR(50) | NOT NULL |
| Trạng thái (ACTIVE, INACTIVE, SUSPENDED) | start\_date | DATE | NULL | Ngày bắt đầu hợp tác |

## Giao diện Người dùng

### Màn hình Danh sách Đối tác



### Màn hình Form Thêm Đối tác



# Chức năng Quản lý Tài sản

## Khảo sát hệ thống

Chức năng quản lý tài sản cho phép:

* Quản lý thông tin thiết bị/tài sản công ty
* Phân bổ thiết bị cho nhân viên
* Theo dõi trạng thái thiết bị
* Quản lý lịch sử sử dụng

## Kịch bản sử dụng

### Use Case 1: Thêm thiết bị mới

**Mô tả**: HR/Admin thêm thiết bị mới vào hệ thống

**Luồng thực hiện**: 1. Truy cập menu "Thiết bị" 2. Click "Thêm thiết bị" 3. Điền thông tin: - Mã thiết bị (EQ-001) - Tên thiết bị - Loại thiết bị - Số serial - Trạng thái (Có sẵn, Đã phân bổ, Bảo trì, Hỏng, Đã nghỉ hưu) - Ngày mua, Giá mua - Ghi chú 4. Click "Lưu"

**Kết quả**: Thiết bị mới được tạo

### Use Case 2: Phân bổ thiết bị cho nhân viên

**Mô tả**: HR/Admin phân bổ thiết bị cho nhân viên sử dụng

**Luồng thực hiện**: 1. Chọn thiết bị cần phân bổ 2. Click "Phân bổ" 3. Chọn nhân viên từ danh sách 4. Xác nhận phân bổ 5. Hệ thống cập nhật trạng thái thiết bị thành "Đã phân bổ"

**Kết quả**: Thiết bị được phân bổ cho nhân viên

### Use Case 3: Thu hồi thiết bị

**Mô tả**: HR/Admin thu hồi thiết bị từ nhân viên

**Luồng thực hiện**: 1. Chọn thiết bị đã phân bổ 2. Click "Thu hồi" 3. Xác nhận thu hồi 4. Hệ thống cập nhật trạng thái thiết bị thành "Có sẵn"

**Kết quả**: Thiết bị được thu hồi và có thể phân bổ lại

## Biểu đồ Sequence

Failed to generate image: undefined method `to\_sym' for nil:NilClass  
@startuml  
actor "Người dùng (HR/Admin)" as User  
participant "Giao diện Quản lý Thiết bị" as View  
participant "EquipmentController" as Controller  
participant "EquipmentService" as Service  
participant "CSDL (EquipmentRepository)" as DB  
participant "EmployeeRepository" as EmpRepo  
  
== Xem danh sách thiết bị ==  
User -> View: Mở trang Quản lý Thiết bị  
View -> Controller: GET /cms/equipment  
Controller -> Service: Lấy danh sách thiết bị (có lọc)  
Service -> DB: Truy vấn dữ liệu thiết bị với filters  
DB --> Service: Danh sách thiết bị  
Service --> Controller: Danh sách EquipmentResponse  
Controller -> View: Render giao diện danh sách  
User <- View: Hiển thị danh sách thiết bị  
  
== Thêm thiết bị ==  
User -> View: Nhập thông tin thiết bị mới  
View -> Controller: POST /cms/equipment (EquipmentRequest)  
Controller -> Service: Gọi hàm create(request)  
Service -> Service: Kiểm tra và ánh xạ dữ liệu (validate + mapping)  
Service -> DB: Lưu thiết bị mới vào CSDL  
DB --> Service: Trả về đối tượng thiết bị đã lưu  
Service --> Controller: Trả thông báo thành công  
Controller -> View: Reload danh sách + hiển thị thông báo "Thêm mới thành công"  
User <- View: Xem danh sách có thiết bị mới  
  
== Cập nhật thiết bị ==  
User -> View: Chọn chỉnh sửa thông tin thiết bị  
View -> Controller: POST /cms/equipment/{id}/update (EquipmentRequest)  
Controller -> Service: Gọi hàm update(id, request)  
Service -> DB: Tìm thiết bị theo ID  
DB --> Service: Trả về thông tin thiết bị  
Service -> Service: Cập nhật thông tin mới vào entity  
Service -> DB: Ghi lại thông tin đã cập nhật  
DB --> Service: Xác nhận thành công  
Service --> Controller: Thông báo cập nhật thành công  
Controller -> View: Cập nhật lại danh sách  
User <- View: Xem danh sách đã thay đổi  
  
== Phân bổ thiết bị cho nhân viên ==  
User -> View: Chọn thiết bị và nhân viên để phân bổ  
View -> Controller: POST /cms/equipment/{id}/assign (AssignEquipmentRequest)  
Controller -> Service: Gọi hàm assignToEmployee(id, request)  
Service -> DB: Tìm thiết bị theo ID  
DB --> Service: Trả về thiết bị  
Service -> EmpRepo: Tìm nhân viên theo ID  
EmpRepo --> Service: Trả về nhân viên  
Service -> Service: Cập nhật thiết bị (assignedTo, status = ASSIGNED)  
Service -> DB: Lưu thay đổi  
DB --> Service: Xác nhận thành công  
Service --> Controller: Thông báo phân bổ thành công  
Controller -> View: Cập nhật trạng thái thiết bị  
User <- View: Xem thiết bị đã được phân bổ  
  
== Thu hồi thiết bị ==  
User -> View: Chọn thu hồi thiết bị  
View -> Controller: POST /cms/equipment/{id}/unassign  
Controller -> Service: Gọi hàm unassign(id)  
Service -> DB: Tìm thiết bị theo ID  
DB --> Service: Trả về thiết bị  
Service -> Service: Cập nhật thiết bị (assignedTo = null, status = AVAILABLE)  
Service -> DB: Lưu thay đổi  
DB --> Service: Xác nhận thành công  
Service --> Controller: Thông báo thu hồi thành công  
Controller -> View: Cập nhật trạng thái thiết bị  
User <- View: Xem thiết bị đã được thu hồi  
  
== Xóa thiết bị ==  
User -> View: Chọn xóa thiết bị  
View -> Controller: POST /cms/equipment/{id}/delete  
Controller -> Service: Gọi hàm delete(id)  
Service -> DB: Xóa bản ghi thiết bị  
DB --> Service: Xác nhận xóa thành công  
Service --> Controller: Thông báo xóa thành công  
Controller -> View: Cập nhật lại danh sách thiết bị  
User <- View: Danh sách sau khi xóa  
  
== Lọc thiết bị ==  
User -> View: Nhập điều kiện lọc (tên, trạng thái, loại)  
View -> Controller: GET /cms/equipment?filters  
Controller -> Service: Gọi hàm findByFilters(filters)  
Service -> DB: Truy vấn với điều kiện lọc  
DB --> Service: Kết quả lọc  
Service --> Controller: Danh sách thiết bị đã lọc  
Controller -> View: Hiển thị kết quả lọc  
User <- View: Danh sách thiết bị theo điều kiện  
@enduml

## Biểu đồ Thực thể (ERD)

Failed to generate image: undefined method `to\_sym' for nil:NilClass  
@startuml  
!define TABLE(name,desc) class name as "desc" << (T,#FFAAAA) >>  
!define PK(x) <b><color:#b8861b><&key></color> x</b>  
!define FK(x) <color:#aaaaaa></color> x  
!define NN(x) <color:#ef4444></color> x  
!define UQ(x) <color:#3b82f6></color> x  
  
package "Equipment Management" {  
 TABLE(equipment, "equipment") {  
 PK(id) BIGINT AUTO\_INCREMENT  
 UQ(code) VARCHAR(50) NOT NULL  
 NN(name) VARCHAR(150) NOT NULL  
 type VARCHAR(100)  
 serial\_number VARCHAR(100)  
 NN(status) VARCHAR(50) NOT NULL  
 purchase\_date DATE  
 purchase\_price DECIMAL(12,2)  
 notes VARCHAR(500)  
 FK(assigned\_to\_employee\_id) BIGINT  
 }  
  
 TABLE(employees, "employees") {  
 PK(id) BIGINT AUTO\_INCREMENT  
 UQ(code) VARCHAR(50) NOT NULL  
 NN(full\_name) VARCHAR(150) NOT NULL  
 UQ(email) VARCHAR(150) NOT NULL  
 phone VARCHAR(30)  
 department VARCHAR(100)  
 position VARCHAR(100)  
 NN(hire\_date) DATE NOT NULL  
 termination\_date DATE  
 employment\_status VARCHAR(50)  
 base\_salary DECIMAL(12,2)  
 FK(manager\_id) BIGINT  
 FK(user\_id) BIGINT  
 }  
}  
  
' Relationships  
equipment ||--o| employees : "assigned\_to\_employee\_id"  
  
note right of equipment  
 \*\*Equipment Entity\*\*  
 - Quản lý tài sản/thiết bị công ty  
 - Hỗ trợ phân bổ cho nhân viên  
 - Theo dõi trạng thái thiết bị  
 - Lưu trữ thông tin mua sắm và bảo trì  
end note  
  
note right of employees  
 \*\*Employee Reference\*\*  
 - Nhân viên được phân bổ thiết bị  
 - Liên kết 1-nhiều với equipment  
 - Hỗ trợ báo cáo thiết bị theo nhân viên  
end note  
  
note bottom of equipment  
 \*\*Equipment Status:\*\*  
 - AVAILABLE: Có sẵn  
 - ASSIGNED: Đã phân bổ  
 - MAINTENANCE: Bảo trì  
 - BROKEN: Hỏng  
 - RETIRED: Đã nghỉ hưu  
end note  
@enduml

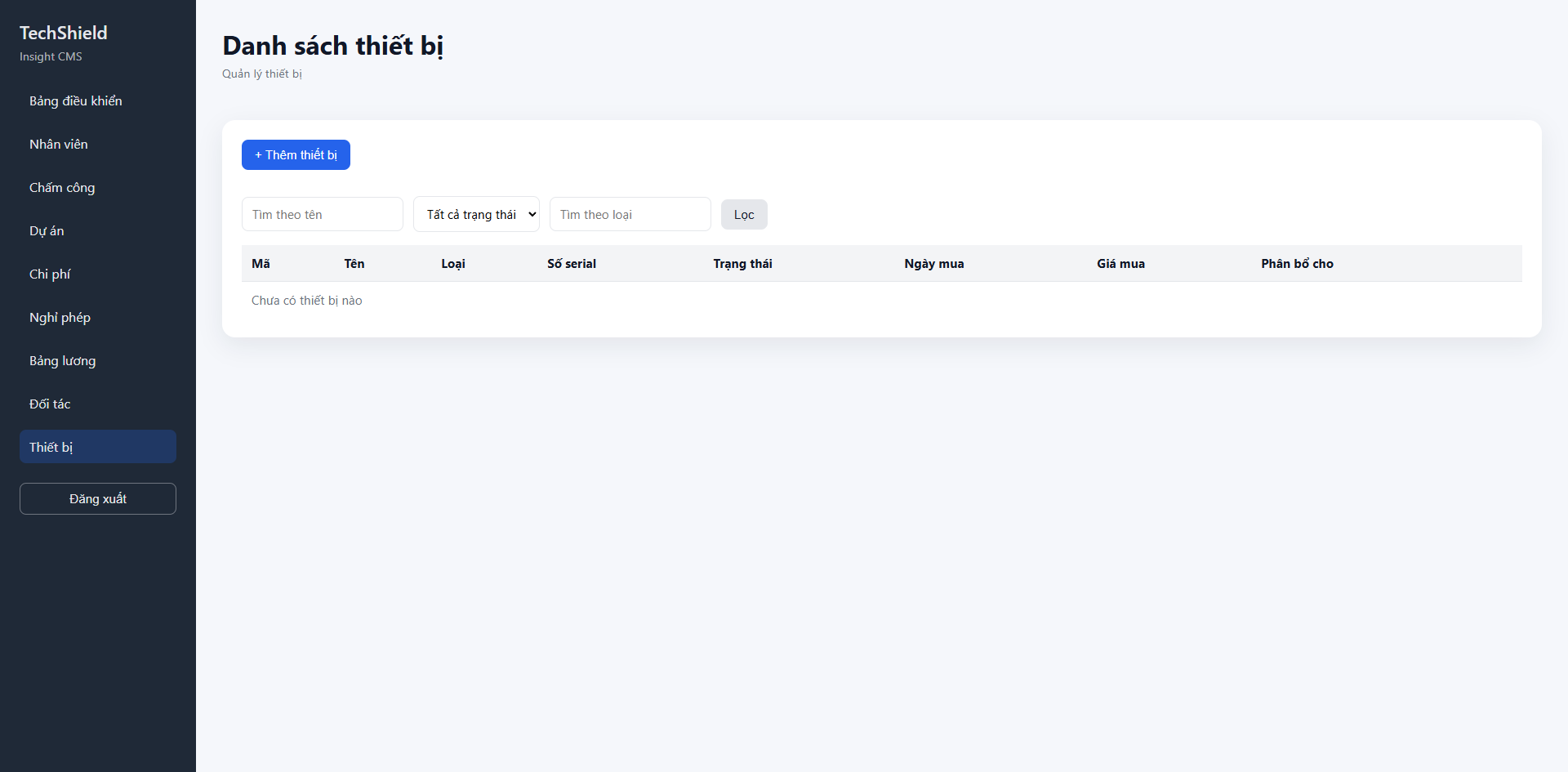
## Thiết kế Cơ sở Dữ liệu

### Bảng equipment

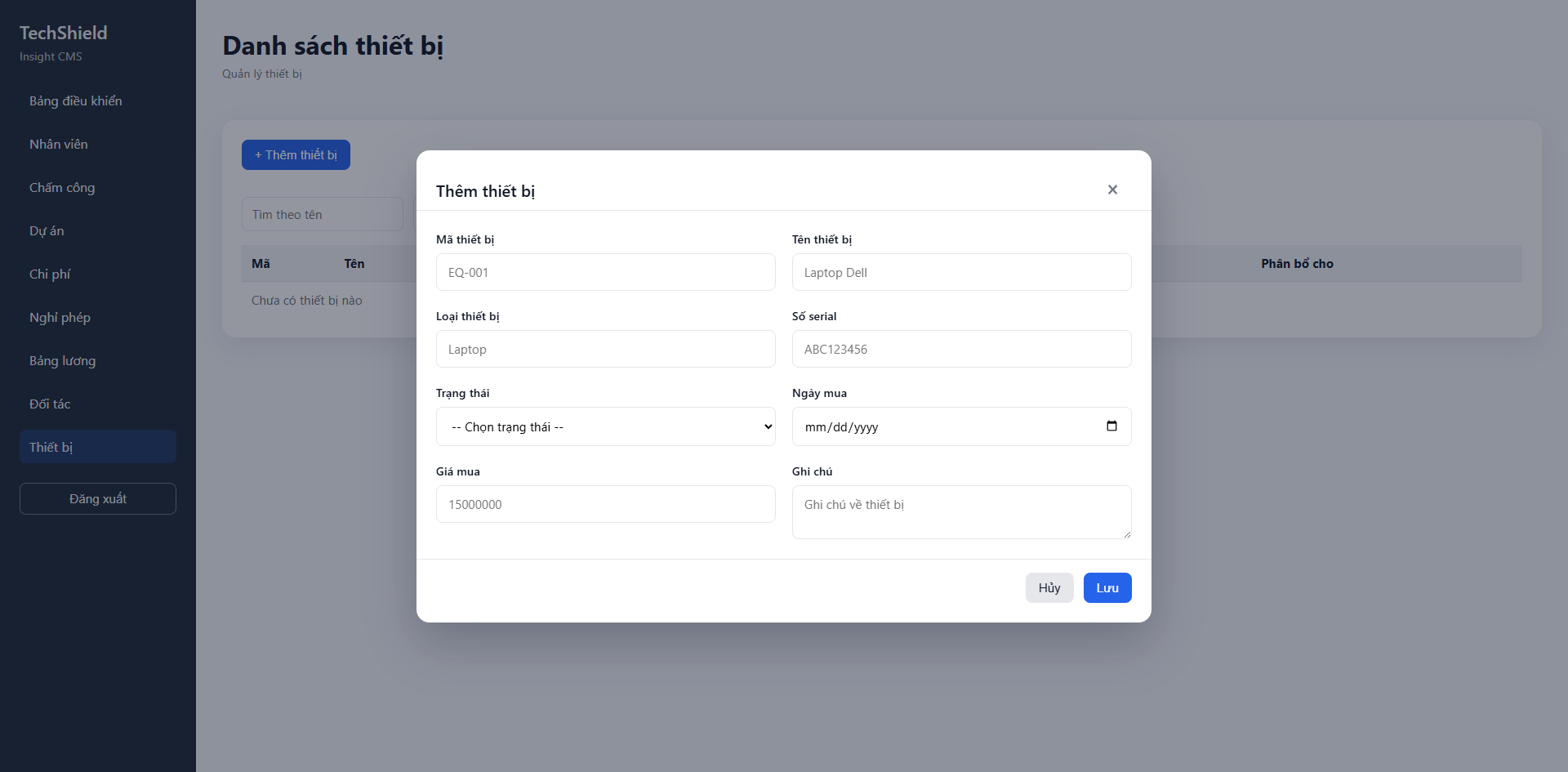
| Cột | Kiểu dữ liệu | Ràng buộc | Mô tả | id |
| --- | --- | --- | --- | --- |
| BIGINT AUTO\_INCREMENT | PK | Khóa chính | code | VARCHAR(50) |
| NOT NULL, UNIQUE | Mã thiết bị (EQ-001) | name | VARCHAR(150) | NOT NULL |
| Tên thiết bị | type | VARCHAR(100) | NULL | Loại thiết bị |
| serial\_number | VARCHAR(100) | NULL | Số serial | status |
| VARCHAR(50) | NOT NULL | Trạng thái (AVAILABLE, ASSIGNED, MAINTENANCE, BROKEN, RETIRED) | purchase\_date | DATE |
| NULL | Ngày mua | purchase\_price | DECIMAL(12,2) | NULL |
| Giá mua | notes | VARCHAR(500) | NULL | Ghi chú |

## Giao diện Người dùng

### Màn hình Danh sách Thiết bị



### Màn hình Form Thêm Thiết bị



# Kết luận

Tài liệu này đã mô tả chi tiết 4 chức năng cốt lõi của hệ thống HRM InShield:

* **Quản lý Nhân viên**: Quản lý thông tin nhân viên, tạo tài khoản, phân quyền
* **Chấm công**: Ghi nhận thời gian làm việc, tính toán giờ công
* **Quản lý Đối tác**: Quản lý thông tin đối tác, phân loại và trạng thái
* **Quản lý Tài sản**: Quản lý thiết bị, phân bổ và thu hồi

Mỗi chức năng được thiết kế với kiến trúc rõ ràng, luồng xử lý logic và giao diện thân thiện với người dùng. Hệ thống sử dụng các công nghệ hiện đại và tuân thủ các nguyên tắc thiết kế phần mềm tốt.

## Công nghệ và Kiến trúc

Hệ thống được xây dựng trên nền tảng Spring Boot với:

* **Backend**: Spring Boot, Spring Security, Spring Data JPA
* **Frontend**: Thymeleaf, Bootstrap
* **Database**: MariaDB với Liquibase migration
* **Security**: Role-based access control

## Hướng phát triển

Hệ thống có thể được mở rộng với các chức năng:

* Báo cáo và thống kê nâng cao
* Tích hợp API với hệ thống khác
* Mobile app cho nhân viên
* Workflow approval cho các quy trình
* Analytics và Business Intelligence