**TÀI LIỆU THIẾT KẾ CẤP CAO**

**High Level Design Document - HLDD**

**Phần mềm quản lý cơ sở  
 dữ liệu của trang web   
Hancity**

**Phiên bản: 1.1**

**Được hướng dẫn bởi:** *Ths. Vo Hoang Quan*

**Được soạn bởi:** *Group 12*

**TON DUC THANG UNIVERSITY  
Thành Phố Hồ Chí Minh, ngày 02 tháng 04 năm 2024**

**MỤC LỤC**

[**1. Giới thiệu: 3**](#_heading=h.3znysh7)

[1.1. Mục đích: 3](#_heading=h.2et92p0)

[1.2. Phạm vi và ràng buộc hệ thống: 3](#_heading=h.tyjcwt)

**[2. Kiến trúc hệ thống: 4](#_heading=h.3dy6vkm)**

[2.1. Lớp presentation (UI): 5](#_heading=h.1t3h5sf)

[2.2. Lớp business: 6](#_heading=h.4d34og8)

[2.3. Lớp data access (DAO): 6](#_heading=h.2s8eyo1)

[2.4. Lớp database (cơ sở dữ liệu): 6](#_heading=h.17dp8vu)

**[3. Góc nhìn logic: 7](#_heading=h.3rdcrjn)**

[3.1. Presentation: 7](#_heading=h.26in1rg)

[3.2. Business: 7](#_heading=h.lnxbz9)

[3.3. Data access: 7](#_heading=h.35nkun2)

[3.4. Database: 8](#_heading=h.1ksv4uv)

[3.5. Value object: 8](#_heading=h.44sinio)

[3.6. Commons: 8](#_heading=h.2jxsxqh)

**[4. Design pattern và framework: 9](#_heading=h.z337ya)**

[4.1. Mẫu thiết kế: 9](#_heading=h.3j2qqm3)

[4.1.1. Mẫu thiết kế tạo đối tượng (Creational design pattern): 9](#_heading=h.1y810tw)

[4.1.2. Mẫu thiết kế tạo cấu trúc (Structural design pattern): 9](#_heading=h.4i7ojhp)

[4.1.3. Mẫu thiết kế hành vi: 9](#_heading=h.2xcytpi)

[4.2. Framework Dot Net 4.7.2: 10](#_heading=h.1ci93xb)

# Giới thiệu:

Phần mềm được thiết kế để đáp ứng nhu cầu quản lý dữ liệu hiệu quả và an toàn cho trang web của công ty. Hệ thống này cung cấp một tập hợp các chức năng giúp quản lý các thao tác CRUD (Create, Read, Update, Delete) trên dữ liệu, cũng như bảo mật dữ liệu, quản lý nhân viên, hợp đồng và cung cấp các báo cáo thống kê.

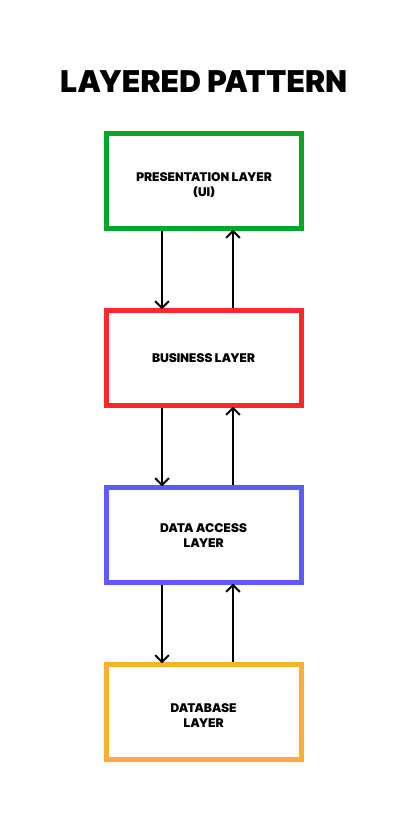
## Mục đích:

* Cung cấp một bản thiết kế chi tiết và toàn diện cho phần mềm quản lý cơ sở dữ liệu (DBManager).
* Đảm bảo DBManager đáp ứng đầy đủ các yêu cầu nghiệp vụ và kỹ thuật.
* Làm rõ tầm nhìn, mục tiêu và chiến lược cho việc phát triển DBManager.
* Hướng dẫn các bên liên quan (stakeholders) trong quá trình phát triển và triển khai DBManager.

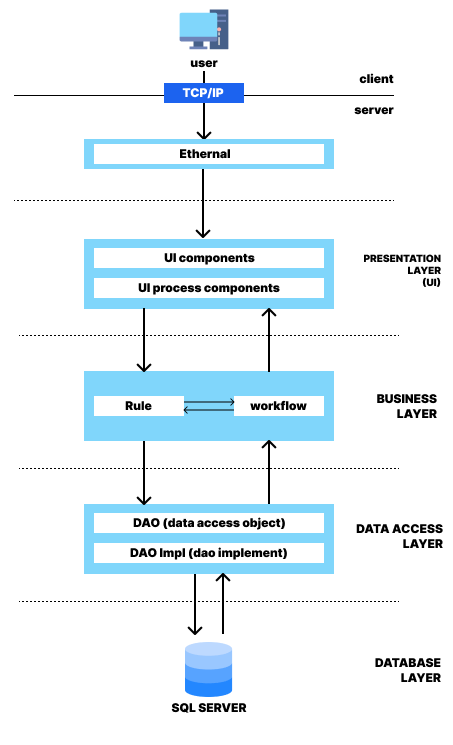
## Phạm vi và ràng buộc hệ thống:

* Tài liệu thiết kế cấp cao giúp xác định ranh giới về chức năng, phi chức năng, dữ liệu, người dùng, nền tảng hệ thống và nhiều yếu tố khác.
* Tài liệu này cung cấp thông tin về cách thức hoạt động và phương hướng thiết kế của phần mềm quản lý cơ sở dữ liệu (DBManager).
* Tài liệu liệt kê các ràng buộc về giới hạn phần cứng, phần mềm và công cụ khác mà người dùng cuối phải tuân thủ để phần mềm được hoạt động với hiệu quả tối đa.

# Kiến trúc hệ thống:

****

Hình 1 - Mẫu thiết kế kiến trúc hệ thống

****

Hình 2 - Kiến trúc hệ thống của DBManager

* Hệ thống quản lý cơ sở dữ liệu (DBManager) được thiết kế theo kiến trúc phân tầng (Layered pattern). Mô hình thiết kế phần mềm này vô cùng phổ biến, nó chia ứng dụng thành các lớp riêng biệt, mỗi lớp có chức năng và trách nhiệm cụ thể.
* Phần mềm sử dụng ứng dụng WINFORM với framework .NET version 4.7.2 và SQL server để hiện thực hệ thống.

## Lớp presentation (UI):

* Cung cấp giao diện cho người dùng tương tác với hệ thống (người dùng cuối - end-user).
* Hiển thị dữ liệu và thông tin, đồng thời thu thập thông tin từ người dùng.
* Gửi yêu cầu (request) từ các tương tác của người dùng với hệ thống đến lớp nghiệp vụ (business).

## Lớp business:

* Xử lý các yêu cầu từ lớp giao diện người dùng (UI - presentation).
* Thực hiện các thao tác CRUD (Create, read, update, delete) trên dữ liệu.
* Áp dụng các quy tắc nghiệp vụ được quy định trước đó của hệ thống.
* Gửi yêu cầu (request) từ các tương tác của người dùng cần truy cập đến dữ liệu đến lớp truy cập dữ liệu (data access).

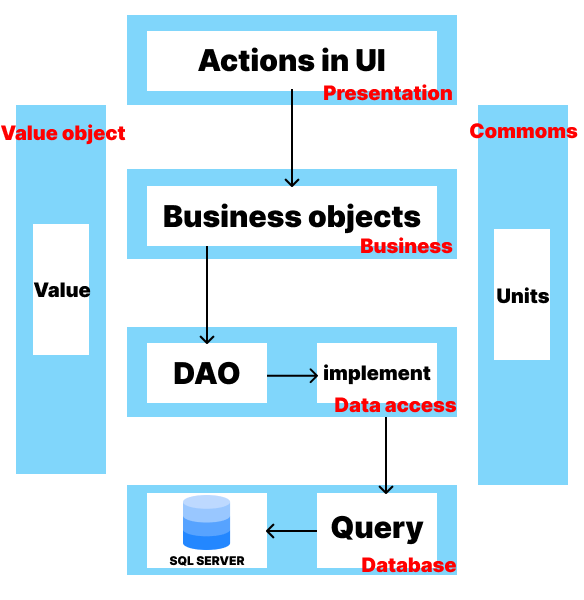
## Lớp data access (DAO):

* Truy cập và thao tác dữ liệu trong cơ sở dữ liệu.
* Sử dụng các câu lệnh truy vấn dữ liệu để thực hiện các yêu cầu và CRUD.
* Cung cấp dữ liệu cho lớp nghiệp vụ.

## Lớp database (cơ sở dữ liệu):

* Lưu trữ dữ liệu của hệ thống.
* Cung cấp các dịch vụ truy cập dữ liệu của lớp truy cập dữ liệu (data access).

# Góc nhìn logic:

****

Hình 3 - Sơ đồ thành phần dựa trên kiến trúc hệ thống

## Presentation:

Quy trình trên hiện thực các thao tác của người dùng cuối ở lớp UI.

## Business:

Lớp business sẽ xác định các đối tượng thuộc các lớp, đồng thời xác định các phương thức trong có trong lớp để chỉ ra các đối tượng nghiệp vụ liên quan từ đó thực thi các yêu cầu từ người dùng.

## Data access:

Các đối tượng và phương thức được xác định từ lớp nghiệp vụ, lớp truy cập dữ liệu sẽ từ đó xác định các lớp chứa đối tượng, phương thức và lớp mà các đối tượng, phương thức truy cập đến. Từ đó định nghĩa cách thức truy cập dữ liệu với các tham số và trả về các giá trị phù hợp hoặc trả về các trường hợp ngoại lệ tránh lỗi cho toàn bộ chương trình.

Sau đó các đối tượng ánh xạ dữ liệu đến cơ sở dữ liệu và nhận lại từ cơ sở dữ liệu.

## Database:

Từ việc xác định cách thức truy cập dữ liệu, lớp này sẽ thực hiện các thao tác như: kết nối, truy vấn, thêm, sửa, xóa dữ liệu hoặc thực thi các procedure, transaction.

## Value object:

Đây là một kiểu dữ liệu không thay đổi (không có trạng thái) trong hệ thống dùng làm các thuộc tính, hàm, phương thức mặc định, đơn giản để tránh lỗi.

## Commons:

Hệ thống sử dụng một số thư viện (API) hỗ trợ các chức năng trong phần mềm như: AutoMapper, Apache commons collections,..

Bằng cách sử dụng những mã nguồn mở, chúng tôi tiết kiệm được thời gian và công sức nhưng lại nâng cao chất lượng và mở rộng hệ thống.

# Design pattern và framework:

## Mẫu thiết kế:

### Mẫu thiết kế tạo đối tượng (Creational design pattern):

Các đối tượng trong hệ thống được khởi tạo dưới dạng singleton(được chỉ định tại một lớp) và abstract factory(cung cấp giao diện để tạo ra các đối tượng, không cần chỉ định lớp cụ thể).

### Mẫu thiết kế tạo cấu trúc (Structural design pattern):

Mô hình hóa các đối tượng và lớp của chúng được linh hoạt và dễ dàng mở rộng hơn, chúng tôi sử dụng mẫu Adapter (các đối tượng không tương thích có thể làm việc cùng nhau), Decorator (Thêm các chức năng mới mà không cần thay đổi cấu trúc của đối tượng) và Composite (Cho phép xây dựng cấu trúc phức tạp từ các đối tượng đơn giản).

### Mẫu thiết kế hành vi:

Thể hiện các tương tác và trao đổi dữ liệu của các đối tượng, chúng tôi sử dụng mẫu Observer (1 đối tượng theo dõi trạng thái của 1 đối tượng khác và thông báo khi có sự thay đổi), mẫu Command (Chuyển đổi yêu cầu/ hoạt động thành 1 đối tượng độc lập => phân tách người gửi yêu cầu và người thực hiện yêu cầu).

## Framework Dot Net 4.7.2:

Microsoft .NET Framework 4.7.2 là phiên bản cập nhật của .NET Framework 4.7, cung cấp nhiều tính năng mới và cải tiến cho các nhà phát triển ứng dụng Windows. Đặc biệt, hệ thống quản lý cơ sở dữ liệu (DBManager) sử dụng WINFORMS nên framework này hỗ trợ rất hiệu quả.

* Hiệu suất mạnh mẽ:
  + Cải thiện hiệu suất cho các ứng dụng ASP.NET Core và Entity Framework Core.
  + Tối ưu hóa bộ sưu tập và JIT compiler.
  + Hỗ trợ cho .NET Native, giúp tăng tốc độ khởi động và thời gian thực thi cho ứng dụng.
* Tính năng vượt trội:
  + Hỗ trợ cho Windows 10 Fall Creators Update (phiên bản 1709).
  + Cải thiện cho Windows Presentation Foundation (WPF), Windows Forms và ASP.NET.
  + Hỗ trợ cho .NET Standard 2.0, cho phép phát triển các thư viện .NET di động có thể sử dụng trên nhiều nền tảng khác nhau.
* Khả năng tương thích cao:
  + Tương thích ngược với các phiên bản .NET Framework trước đây.
  + Hỗ trợ cho nhiều ngôn ngữ lập trình, bao gồm **C#**, F# và Visual Basic.
* Lợi ích:
  + Giúp phát triển xây dựng các ứng dụng Windows hiệu quả và mạnh mẽ hơn.
  + Tăng tốc độ phát triển ứng dụng.
  + Cung cấp khả năng tương thích cao với các phiên bản .NET Framework trước đây.