***Phần mềm quản lý hiệu sách***

RMS High Level Design

**Học Phần: CÔNG NGHỆ PHẦN MỀM**

**Nhóm:19Nh14**

**Team 8: Phan Văn Bình**

**Nguyễn Thanh Cơ**

**Lê Văn Huy**

**Võ Thành Vĩ**

**Lê Thanh Long**

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Revision Details** | **Author** | **Date** |
| 1.0 | Creation | XXX | 1/3/2021 |
| 1.01 | Chọn Đề Tàì | XX | 10/3/2021 |
| 1.02 | Xác định yêu cầu, mục tiêu, quy mô đồ án | XX | 16/3/2021 |
| 1.03 | Hoàn thành Project Description | XX | 19/3/2021 |
| 1.04 | Xác định Usecase, Class Diagram | XX | 25/3/2021 |
| 1.05 | Hoàn thành đặc tả SRS | XX | 13/4/2021 |
| 1.06 | Tìm hiểu những app hiện có trên thị trường cùng lĩnh vực | XX | 15/4//2021 |
| 1.07 | Xác định Activity Diagram, Sequence Diagram | XX | 10/4/2021 |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 1.08 | Hoàn thiện Database | XX | 7/4/2021 |
| 1.09 | Hoàn thiện RMS-DD | XX | 15/4/2021 |

MỤC LỤC

[**1.Lời nói đầu** 5](#_Toc20076)

[1.1 Mục đích 5](#_Toc20077)

[1.2 Định nghĩa và từ viết tắt 5](#_Toc20078)

[1.3 Tài liệu tham khảo 5](#_Toc20079)

[**2.Mô tả kiến trúc** 6](#_Toc20080)

[2.1 Tầng giao diện: 7](#_Toc20081)

[2.2 Tầng trung gian 7](#_Toc20082)

[2.3 Tầng dữ liệu 8](#_Toc20083)

[**3.UseCase View** 8](#_Toc20084)

[**4.Localview** 9](#_Toc20085)

[4.1 Overview 9](#_Toc20086)

[4.2 Architecturally Significant Design Packages/Components 9](#_Toc20087)

[4.2.1 JAVA01 presentation (JSPs, Actions) 10](#_Toc20088)

[4.2.2 Business Object 10](#_Toc20089)

[4.2.3 Value Objects 10](#_Toc20090)

[4.2.4 DAO 10](#_Toc20091)

[4.2.5 Hibernate Beans 10](#_Toc20092)

[4.2.6 Util 11](#_Toc20093)

[**5.Phụ lục:** 11](#_Toc20094)

# 1.Lời nói đầu

## 1.1 Mục đích

Tài liệu này cung cấp một cái nhìn tổng quan toàn diện về kiến trúc của Phần mềm quản lý đặt sân thể thao. Với mục đích nắm bắt và truyền đạt các quyết định kiến trúc quan trọng đã được thực hiện về hệ thống, tài liệu đóng vai trò là cầu nối giữa các yêu cầu phần mềm và thiết kế chi tiết của StadiumManagement; nó cũng sẽ giúp các kiến trúc sư phần mềm đảm bảo rằng hệ thống được xây dựng sẽ đáp ứng nhu cầu của người dùng về chức năng (chế độ xem ca sử dụng đã chọn), về nền tảng và công nghệ (chế độ xem logic và triển khai).

## 1.2 Định nghĩa và từ viết tắt

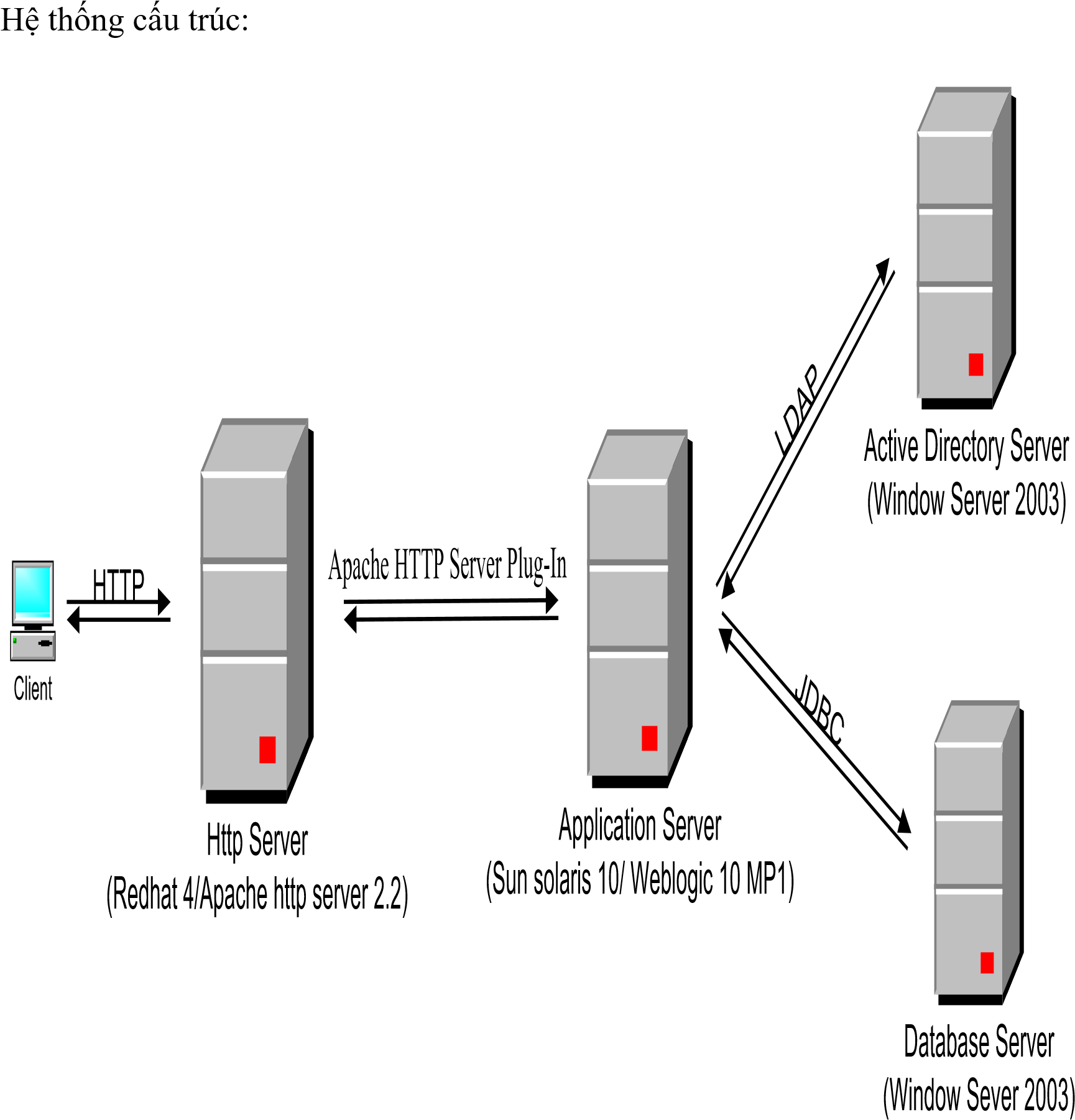
## 

|  |  |  |
| --- | --- | --- |
| STT |  | Mô tả |
| **Từ viết tắt** |
|  |
| 1 | CS | Đuôi mở rộng mã nguồn C# |
| 2 | 3-Layer | Mô hình 3 lớp trong phát triển ứng dụng |
| 3 | DAL | Data Access Layer – Tầng kết nối và xử lý cơ sở dữ liệu |
| 4 | BUS | Business Layer – Tầng trung gian kết nối Tầng DAL với tầng GUI |
| 5 | GUI | Tầng giao diện hiển thị của ứng dụng |
| 6 | Model | Các lớp(Class) mô tả thực thể ứng với các bảng trong Cơ sở dữ liệu |
| 7 | DAO | Data Access Object, this object is responsible for attaching to a system, extracting some information, based on specific requirements, and creating a value object. |
| 8 | EF6 | Entity Framework 6 – ORM (Object Relational Mapping)  Thư viện giúp việc truy xuất và xử lý dữ liệu từ Cơ sở dữ liệu 1 cách dễ dàng. Gọi là EF6 để phân biệt với Entity Framework Core |
| 9 | DI | Dependency Injection  Design Pattern được thiết kế với mục đích ngăn chặn sự phụ thuộc giữa các class, để khiến cho code dễ hiểu hơn, trực quan hơn, nhằm phục vụ cho mục đích bảo trì và nâng cấp code. |

## 1.3 Tài liệu tham khảo

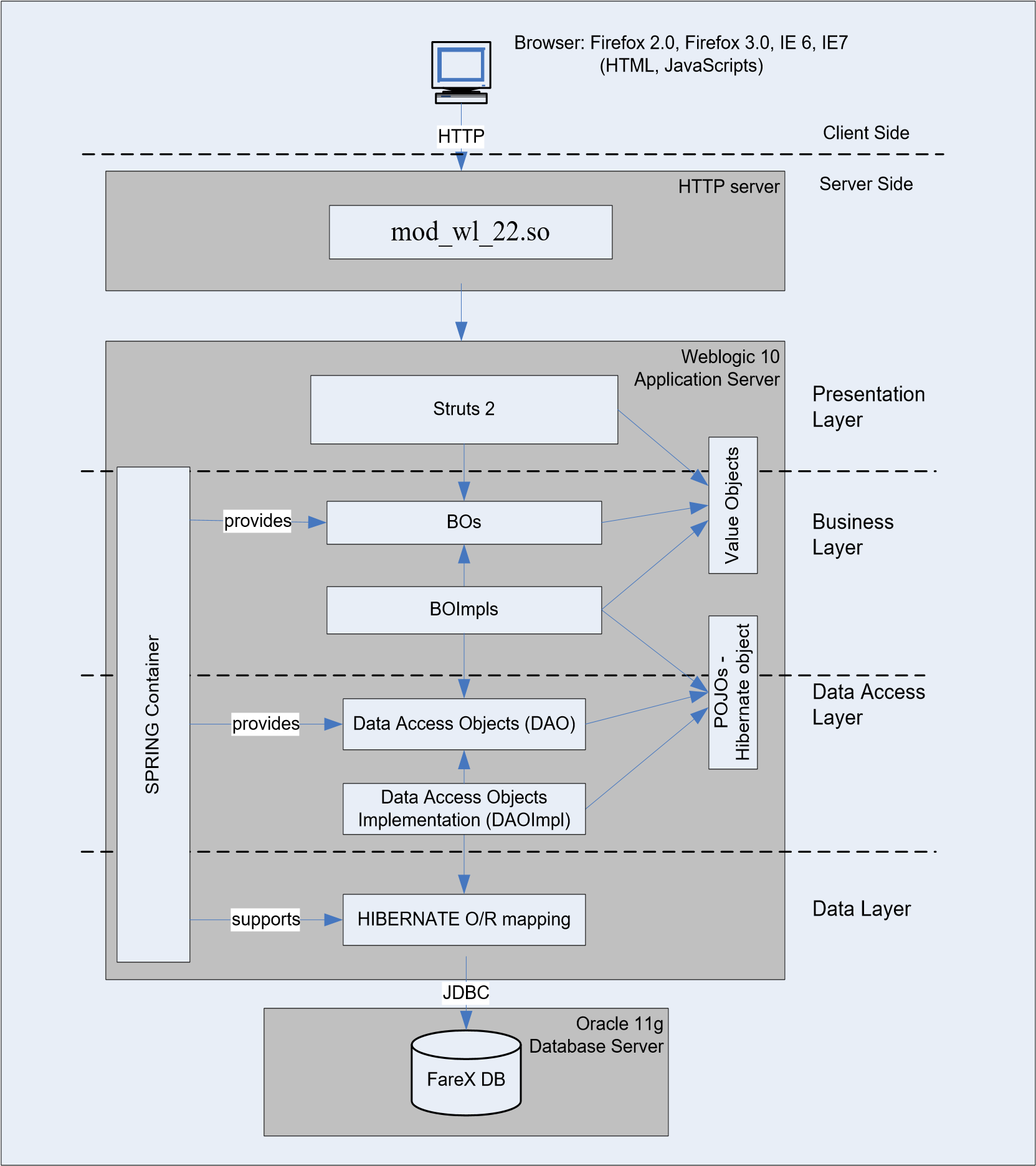
|  |  |
| --- | --- |
| **Document Name** | **Description** |
| Mô hình MVC cơ bản | https://monamedia.co/mvc-la-gi-ung-dung-cua-mo-hinh-mvc-tronglap-trinh/ |
| Kết nối Database | https://ongthovuive.wordpress.com/2016/09/06/c-ket-noi-co-so-du-lieu-sql/ |
| Sử dụng Git | https://thachpham.com/series/git-co-ban |
| Sử dụng ứng dụng Trello để quản lý dự án | https://trello.com/b/9mWcqTWt/project-base-learning-3pbl |

# 2.Mô tả kiến trúc



Biểu đồ sau đây cho thấy các tầng chính trong kiến trúc 3-Layer được đề xuất.

Sơ đồ này cho thấy các lớp chính trong kiến trúc này và cách chúng khớp với nhau



## 2.1 Tầng giao diện:

Thông tin dữ liệu của MVC thông qua các thao tác truy vấn như tìm kiếm hoặc sử dụng thông qua các website.

Thông thường, các ứng dụng web sử dụng MVC View như một phần của hệ thống, nơi các thành phần [HTML](https://monamedia.co/html-va-css-la-gi/) được tạo ra. Bên cạnh đó, View cũng có chức năng ghi nhận hoạt động của người dùng để tương tác với Controller. Tuy nhiên, View không có mối quan hệ trực tiếp với Controller, cũng không được lấy dữ liệu từ Controller mà chỉ hiển thị yêu cầu chuyển cho Controller mà thôi.

## - Tầng giao diện chịu trách nhiệm:

+ Quản lý các yêu cầu, phản hồi từ/đến người dùng.

+ Điều khiển hiển thị cho người dùng cuối.

+ Thiết lập 1 mô hình có thể trình bày trong một khung nhìn.

+ Thức hiện xác nhận giao diện người dùng.

+ Cung cấp 1 bộ điều khiển để ủy quyền các cuộc gọi đến các xử lý nghiệp vụ và các quy trình khác.

+ Xử lý ngoại lệ từ các lớp.

## 2.2 Tầng trung gian

Tầng này quản lý các quy tắc và logic xử lý nghiệp vụ

* Xử lý logic nghiệp vụ của ứng dụng và xác định nghiệp vụ.

- Quản lý các giao dịch giữa tầng View và Model

- Cho phép giao diện tương tác với các lớp khác.

* Quản lý sự phụ thuộc giữa các đối tượng cấp nghiệp vụ.
* Thêm tính linh hoạt giữa lớp View và Model mà không thông qua tiếp xúc trực tiếp với nhau.
* Quản lý việc thực thi từ lớp nghiệp vụ đến lớp bền vững.

## 2.3 Tầng dữ liệu

Tầng này quản lý quyền truy cập vào cơ sở dữ liệu và thông qua Entity Framework 6 để lấy ra dữ liệu của ứng dụng .

Lý do chính để tách quyền truy cập dữ liệu khỏi phần còn lại của ứng dụng là việc chuyển đổi nguồn dữ liệu và chia sẻ đối tượng truy cập dữ liệu (Model) giữa các ứng dụng dễ dàng hơn

* Lớp này quản lý việc đọc, ghi, cập nhật và xóa dữ liệu được lưu trữ trong cơ sở dữ liệu
* Entity Framework 6 là một ORM mạnh mẽ để thực hiện việc Mapping dữ liệu từ Cơ sở dữ liệu vào các Model của ứng dụng để dễ dàng truy cập và thực hiện các thao tác CRUD (Create-Read-Update-Delete). Giúp việc phát triển ứng dụng trở nên nhanh chóng, trực quan và quen thuộc với Lập trình hướng đối tượng.

# 3.UseCase View

Thiếu…..

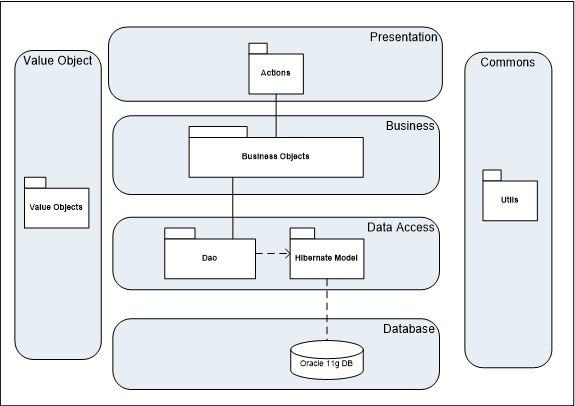
# 4.Localview

## 4.1 Overview

## 

Phần này mô tả các gói / thành phần quan trọng trong kiến trúc của ứng dụng quản lý hiệu sách với thiết kế kiến trúc cũng như trách nhiệm của chúng.

## 4.2 Architecturally Significant Design Packages/Components



### 4.2.1 JAVA01 presentation (JSPs, Actions)

Gói này bao gồm việc triển khai cho Lớp trình bày để xử lý việc hiển thị cho người dùng cuối.

### 4.2.2 Business Object

Tên gói: sg.com.customer.JAVA01.service

Gói này bao gồm việc thực hiện các đối tượng kinh doanh. Lớp Business Object (BO) được sử dụng để thực hiện các hoạt động business. Lớp Đối tượng nghiệp vụ sẽ truy cập DAO bằng cách triển khai mẫu Bộ phun phụ thuộc. Các giao dịch nên được quản lý trong lớp nghiệp vụ này.

* POJOs: Tất cả các đối tượng nghiệp vụ sẽ là POJO và được tạo bằng Spring.
* Phân định giao dịch Spring: Sử dụng quản lý giao dịch khai báo trọng lượng nhẹ của Spring, nơi có thể khai báo các cài đặt giao dịch trên toàn hệ thống. tức là đối với tất cả các phương thức đối tượng nghiệp vụ bắt đầu bằng ‘insert’ sẽ bắt đầu giao dịch nhưng đối với tất cả các phương thức đối tượng nghiệp vụ bắt đầu bằng ‘find’ sẽ không bắt đầu giao dịch. Điều này rất linh hoạt và sẽ được áp dụng bằng Hibernate.

### 4.2.3 Value Objects

Tên gói: sg.com.customer.JAVA01.valueobjects

Đối tượng giá trị là lớp Java, chứa các cấu trúc nhẹ cho thông tin kinh doanh liên

quan. khi được gọi là các đối tượng truyền dữ liệu. Đối tượng giá trị (VO) là một đối tượng nhẹ, có thể tuần tự hóa, cấu trúc các nhóm mục dữ liệu thành một cấu trúc logic duy nhất. (Các đối tượng giá trị luôn triển khai java.io.Serializable).

* Một VO nhằm mục đích giảm thiểu lưu lượng mạng giữa các hạt doanh nghiệp và người gọi của chúng (vì mỗi đối số được truyền sẽ bắt đầu một quá trình truyền mạng).
* Một VO được thiết kế để cải thiện hiệu suất của các bean doanh nghiệp bằng cách giảm thiểu số lượng các đối số của phương thức, và do đó, việc truyền mạng, cần thiết để gọi chúng.
* Ngoài ra, VO rất hữu ích trong giao tiếp giữa tất cả các lớp của ứng dụng

### 4.2.4 DAO

Tên gói: sg.com.customer.JAVA).dao

Gói này bao gồm việc triển khai DAO Java pattern. Sử dụng mẫu thiết kế DAO ở đây để làm cho ứng dụng linh hoạt hơn để truy cập cơ sở dữ liệu. DAO bao gồm các chức năng cơ bản để làm việc với cơ sở dữ liệu: chọn, chèn, cập nhật, xóa.

### 4.2.5 Hibernate Beans

Tên gói: sg.com.customer.JAVA01.model

Gói này bao gồm việc triển khai đối tượng Hibernate - ánh xạ từ cơ sở dữ liệu quan hệ sang các đối tượng java.

Hibernate cung cấp một tập hợp các tính năng phong phú để nhanh chóng xây dựng các ứng dụng doanh nghiệp hiệu suất cao, có khả năng mở rộng và bảo trì.

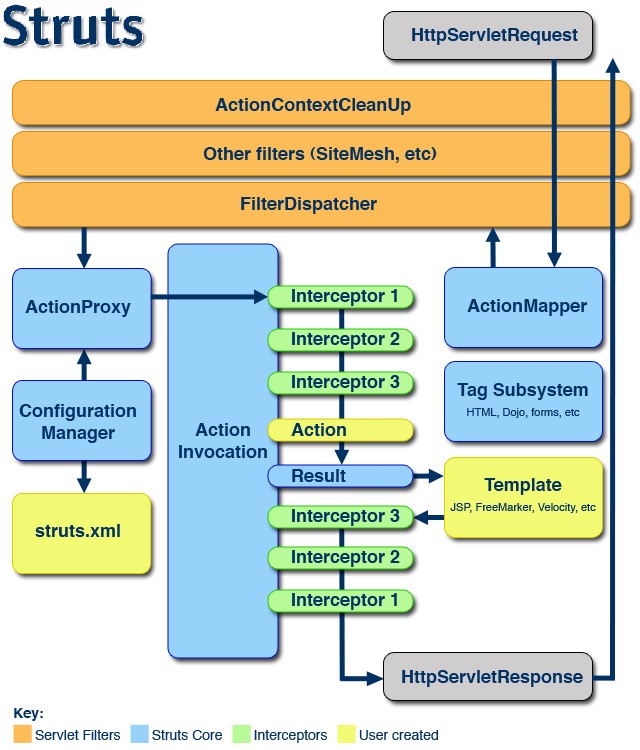
### 4.2.6 Util

Tên gói: sg.com.customer.JAVA01.util

This package includes all utilities Java classes will be wisely used in the modules.

# 5.Phụ lục:

Danh sách các mẫu thiết kế và khuôn khổ được sử dụng: + Struts2 framework: Dưới đây là hệ thống kiến trúc



+SPRING Framework

Chi tiết: [**http://www.springframework.org/**](http://www.springframework.org/)

Hệ thống kiến trúc:

