Mô hình 3 lớp

Lý do sử dụng

Mô hình ba lớp giúp ta tách các phần code về giao diện, xử lý, truy vấn thành các phần riêng biệt trong ứng dụng giúp lập trình viên dễ dàng quản lý code, kiểm tra code, khi có lỗi xảy ra thì có thể khoanh vùng lỗi, dễ bảo trì, khi cần thay đổi giao diện hoặc câu lệnh truy vấn nào đó cũng không ảnh hưởng đến phần còn lại.

Front Controller Pattern

Nơi sử dụng

Điều phối chuyển giữa các màn hình (Form)

Folder frontController (Dispatacher, FrontController)

Lý do sử dụng

Front Controller Pattern là mô hình giúp ta xử lý các tiền xử lý và hậu xử lý trước khi điều hướng đến một nơi nào đó, là một nơi tập trung để quản lý các điều hướng đến các Form trong ứng dụng, đồng thời giúp thuận tiện trong việc thay đổi code, thêm code trước và sau khi điều hướng đến Form khác, như việc: gọi hủy Form trước, mở Form sau hoặc khi muốn thêm một hành động nào đó trước khi chuyển Form thì ta chỉ cần chỉnh sửa tại một nơi, không cần phức tạp chỉnh sửa ở từng nơi.

Template method, strategy pattern

Nơi sử dụng

Template method trong thư mục ui/templatePattern

strategyPattern trong thư mục ui/strategyPattern

FormHome: đổi màn hình(đổi chiến lược) khi chạy và template method làm lại hoạt động (các child là các control)

Lý do sử dụng

Strategy Pattern giúp ta thay đổi dạng thuật toán hoặc một hành vi nào đó trong quá trình chạy ứng dụng. Áp dụng vào bài tập, Strategy Pattern có thể thay đổi các UserControl, tạo ra một nơi tập hợp các UserControl để thực hiện một hành vi có ý nghĩa chung nhưng xử lý khác nhau trên mỗi UserControl, việc này giúp bảo trì code được nhanh chóng và độc lập hơn vì chỉ chỉnh sửa trên một UserControl sẽ không ảnh hưởng đến những UserControl khác.

Template Pattern giúp ta tạo nên một bộ khung của một chức năng nào đó và có thể cho lớp con định nghĩa lại một số hành vi trong bộ khung. Mô hình này áp dụng khi thay đổi các UserControl, nó sẽ thực hiện các hành vi trong một bộ khung đã được định nghĩa trước đó và điều đó tránh việc lặp đi lặp lại code ở nhiều nơi, đồng thời đối với việc các lớp con có hành vi khác nhau, nó giúp định nghĩa lại những hành vi riêng biệt đó, tránh ảnh hưởng đến các hành vi có cấu trúc giống nhau giữa lớp con, cũng như việc khởi tạo riêng cho các phương thức gây nên tình trạng dư thừa không đáng có. Về lâu về dài, điều này giúp hạn chế việc thay đổi code ở nhiều nơi mà chỉ cần chỉnh sửa ở một nơi nhất định.

Factory Pattern

Nơi sử dụng

Folder frontController (FormFactory)

Để tạo ra các màn hình để chuyển

Lý do sử dụng

Factory Pattern là mô hình giúp ta tạo ra các đối tượng theo yêu cầu, được áp dụng để tạo ra các đối tượng Form khác nhau phụ thuộc vào Form muốn khởi tạo, đồng thời mẫu thiết kế cũng giúp ta có một cấu trúc code dễ nhìn, dễ quản lý hơn rất nhiều khi muốn tạo ra một đối tượng.

Singleton Pattern:

Nơi sử dụng

Folder helper

@DatabaseMySql

Folder Dao

FrontController

Lý do sử dụng

Singleton Pattern là một trong những mô hình được sử dụng phổ biến nhất, nó giúp cho việc tạo đối tượng chỉ diễn ra đúng một lần từ khi khởi tạo ứng dụng. Mô hình này giúp giữ các instance của những lớp chỉ cần khởi tạo một lần như: lớp Dao, Helper,… điều đó giúp cho việc gọi đến những lớp này dễ dàng và quan trọng hơn là dữ liệu trong những lớp đó sẽ không bị thay đổi khi gọi lại, làm cho quá trình xây dựng ứng dụng được liền mạch và xuyên suốt.

AbstractFactory:

Nơi sử dụng

Folder helper

Folder Dao

@DatabaseFactory, @DatabaseMySql

Lý do sử dụng

Abstract Factory Pattern là mẫu thiết kế giúp quản lý việc trả về các đối tượng theo yêu cầu. Do đó mẫu thiết kế này tạo ra một nơi thực hiện một việc duy nhất là quản lý, tạo và trả về các đối tượng kết nối database, command, param,… Hiện tại ứng dụng chỉ kết nối với một loại cơ sở dữ liệu chính là SQL nhưng sau này nếu kết nối với một loại cơ sở dữ liệu khác thì mô hình vẫn hỗ trợ để có thể gọi các method trả về đối tượng trước đó đã sử dụng bằng cách thêm một lớp con thừa kế lớp cơ sở dữ liệu của lớp cha mà đã định nghĩa các hàm khởi tạo và điều chỉnh các đối tượng được tạo ra tùy thuộc vào cơ sở dữ liệu đó mà không ảnh hưởng đến lớp cơ sở dữ liệu cũ, cũng như chương trình đã được thực hiện trước đó. Mặt khác, ta cũng giảm thiểu đi việc sử dụng if else quá nhiều để tạo các đối tượng ở cơ sở dữ liệu mà ta muốn.