GRUPO IGPA
GENERADOR DE EXÁMENES
PROGRAMACIÓN AVANZADA

UNIVERSIDAD CENTRAL

JUSTIFICACIÓN USO PATRONES DE DISEÑO

PATRONES DE DISEÑO USADOS

Patrones Básicos

Interfaces
Se requiere el uso de este patrón para agrupar comportamientos comunes que puedan generar diferentes clases del modelo definido para el proyecto. Además se busca cumplir con los principios básicos de diseño, particularmente con el principio de inversión de dependencias que se refiere a que los módulos de alto nivel no deben depender de módulos de bajo nivel ambos deben depender de abstracciones.

Abstract Parent Class
Se requiere el uso de este patrón para agrupar los atributos y comportamientos comunes de los diferentes tipos de preguntas, buscando reducir el nivel de redundancia y facilitar el mantenimiento de la aplicación.

Private Methods
Se requiere el uso de este patrón para acceder a las instancias de las clases. Es necesario que cada clase pueda ser modificada, sólo por ella misma.

Accessor Methods
Se requiere el uso de este patrón para definir los métodos de acceso de cada clase para de esta forma obtener los valores de sus instancias que pueden ser requeridos por otras clases.

Constant Data Manager
Se requiere el uso de este patrón para reunir todas las constantes de la aplicación y ganar organización en el modelo.

Inmutable Object
Se requiere el uso de este patrón para asegurar que no se puedan cambiar los valores luego de instanciar la clase materia, ya que esta es la raíz para la selección de los demás parámetros de configuración de los exámenes.

Patrones Creacionales

Singleton
Este patrón se implementa para asegurar que la instancia de la conexión a la base de datos se realice una sola vez, y sus atributos se mantengan actualizados para las clases que intervengan con esta.

Prototype
Se requiere el uso de este patrón con el fin de a partir de una instancia de un examen creado, clonarlo y cambiar sus valor reduciendo el uso de memoria y agilizando la generación de los exámenes. Lo anterior dado que el objetivo de la aplicación es la generación de un número indeterminado de exámenes (conjunto de preguntas con diferentes características particulares, pero guardando una misma estructura general). La creación de cada unos de ellos podría resultar bastante costoso en términos de rendimiento y uso de memoria.

Patrones Estructurales

Flyweight
Dado que los exámenes que se deben generar cumplen con un conjunto de características comunes, este patrón mejorará el rendimiento en uso de memoria en el proceso de creación de los exámenes.

Facade
Se requiere el uso de este patrón para separar los subsistemas web del subsistema cliente. Se crea una fachada para cada clase concreta.

Abstract Factory
Se requiere el uso de este patrón para guardar una relación entre el tipo de pregunta elegido y las características de la pregunta ya que las preguntas que se incluyan en los cuestionarios pueden tener diferente configuración dependiendo del tipo de pregunta elegido, por ejemplo, una pregunta del falso o verdadero, tiene sólo una respuesta, sin embargo una pregunta de selección múltiple puede tener más de una respuesta, así que se requiere un control dentro de la aplicación en el que de acuerdo al tipo de pregunta tenga un determinado número de respuestas posibles además de la complejidad y el porcentaje.

Patrones De Comportamiento

Iterator
Se requiere el uso de este patrón para recorrer la colección de los exámenes generados, compararlos y evaluar que todos sean distintos entre ellos.

Memento
Se requiere el uso de este patrón para guardar los estados validos del conjunto de parámetros seleccionados en la interfaz para generación de los exámenes. De esta forma se podrá volver a los parámetros seleccionados previamente y generar una nueva solicitud para generación de exámenes.

Observer
Se requiere el uso de este patrón para la selección de los parámetros de la configuración de los exámenes. En la interfaz gráfica el primer campo que se debe seleccionar es la materia y basado en una materia seleccionada se cargan los temas, si el usuario cambia la materia elegida, la aplicación debe recargar el control con los nuevos temas correspondientes a la materia seleccionada. Este patrón nos permite informar a las clases afectadas, este cambio para que ellas puedan cargar la nueva información.

Chain of Responsability
Se requiere el uso de este patrón ya que para el proceso de generación de exámenes se deben definir ciertos parámetros para configuración de los mismos. Los parámetros a definir son: Materia, temas de la materia, tipo de pregunta y complejidad. Cada parámetro depende del anterior y se deben realizar en ese orden. De esta forma una vez se elija la materia, la clase materia sabrá que debe continuar con la instanciación de los temas de la materia y de esta forma sucesivamente.

PATRONES NO USADOS

Patrones Básicos

Monitor
No se requiere usar este patrón dado que la aplicación está construida exclusivamente para consultar las preguntas teniendo en cuenta diferentes parámetros. No se va a permitir realizar ninguna modificación los datos mostrados, lo que evita que se puedan generar inconsistencias en los datos mostrados.

Patrones Creacionales

Factory Method
No se requiere el uso de este patrón ya que la creación de los exámenes depende de la instanciación de varias clases y se hace necesario guardar una relación entre estas. Lo que no es permitido por este patrón.

Builder
No se requiere el uso de este patrón ya que aunque este podría solucionar la complejidad en el momento de la creación de los exámenes implementó Abstract Factory para resolver este problema.

Patrones Estructurales

Adapter

No se requiere el uso de este patrón ya que se identifican clases que sean incompatibles y que deban tener relación entre sí.

Decorator
No se requiere el uso de este patrón, dado que en el alcance del proyecto fueron definidas las responsabilidades para cada clase. (No es necesario agregar responsabilidad a alguna de las clases del modelo).

Composite
No se requiere el uso de este patrón ya que dentro del modelo definido para la aplicación no se implementan estructuras complejas.

Patrones De Comportamiento

Visitor
No se requiere el uso de este patrón ya que cada clase tiene sus operaciones definidas, las estructuras de datos usadas son similares. Los comportamientos comunes de las clases están agrupados en una clase llamada Abstract Facade y no son estructuras heterogéneas.

Command
No se requiere el uso de este patrón ya que la cadena de instanciación de las clases para la creación de los exámenes está dada por medio del patrón Cadena de Responsabilidades