Ivan Renteria

Universidad panamericana  campus Aguascalientes

Proyecto Final



# Base de Datos

La Base de Datos que yo decidí utilizar para mi proyecto fue Microsoft SQL Server.

Me decidí por esta ya que es la única que he utilizado, gracias a la materia de Bases de Datos en semestres pasados, por lo que para este proyecto no fue tan complicado.

También fue muy importante que, en ejercicios anteriores (de esta materia) hubo algunos en los que utilizamos Base de Datos, y fue en esta misma en la que trabajé, por lo que de alguna forma estuve reforzando mis conocimientos en SQL Server.

Incluso, para este proyecto, tomé dichos ejercicios para ver mis tablas, procedimientos, etc. Y darme una idea de como hacerlo para este proyecto de Twitter.

# Arquitectura

El patrón de arquitectura que yo decidí utilizar fue el de M-V-C (Modelo – Vista – Controlador).

Desde un principio fue el que mas me gustó (junto con NCapas), por lo tanto, es el que más se me facilita.

La forma de trabajar de este patrón es separar la lógica de la aplicación de la vista en una aplicación.

Es una arquitectura importante puesto que se utiliza tanto en componentes gráficos básicos hasta sistemas empresariales; la mayoría de los frameworks modernos utilizan MVC (o alguna adaptación del MVC) para la arquitectura, entre ellos podemos mencionar a Ruby on Rails, Django, AngularJS y muchos otros más.

# Patrones Utilizados

Al final de mi trabajo, traté de implementar 4 patrones de diseño.

2 de creación, los cuales fueron Singleton y Builder.

**Singleton** es un patrón de diseño que permite restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase solo tenga una instancia y proporcionar un punto de acceso global a ella.

Este patrón fue de mi agrado desde que empezamos a trabajar con él, ya que al trabajar con una Base de Datos es un patrón que te puede facilitar las cosas, además de permanecer ordenado. Desde mi punto de vista es uno de los patrones más sencillos.

**Builder** es un patrón que separa la creación de un objeto complejo de su representación de modo que el mismo proceso de construcción pueda crear representaciones diferentes del mismo objeto.

Básicamente significa que, en lugar de implementar la creación de un objeto en el constructor de la clase, va a haber otras clases encargadas de crear el objeto y asignarle las propiedades iniciales.

Al momento de ver este patrón en clase, lo hicimos tomando el ejemplo de una pizza, dónde para no tener que recordar cada vez que creemos una pizza su tamaño, salsa, etc. Utilizamos el patrón y nos olvidamos de sus parámetros.

Por lo que decidí utilizar este patrón para mi creación de usuarios de Twitter.

Donde sus parámetros van a ser diferentes a cada registro o a cada inicio de sesión.

Los siguientes 2 patrones que traté de implementar son estructurales, los cuales son Facade y Proxy.

**Facade** es un patrón que tiene como propósito proporcionar una interfaz unificada a otro conjunto de interfaces de un subsistema.

Este patrón es utilizado cuando se necesita tener una interfaz limitada pero directa a un subsistema completo, por lo cual creí que podía utilizarlo al inicio de mi proyecto.

Donde tengo un form como mi primera vista, siendo este una interfaz bastante limitada que te llevará directamente a dos opciones más complejas, como lo son el Inicio de Sesión o el sistema de Registro.

El patrón **Proxy** tiene como propósito controlar el acceso al objeto original, lo que le permite realizar algo antes o después de que la solicitud llegue al objeto original.

Este patrón se puede utilizar cuando se necesite una ejecución local de un servicio remoto (proxy remoto). Esto cuando el objeto de servicio se encuentra en un servidor remoto.

Este patrón decidí utilizarlo al momento de llevar a cabo un registro.

Al requerir un campo de “Ciudad” para registrarte, consideré que sería más fácil o al menos más agradable a la vista, que el usuario no tuviera que teclear su lugar de origen, por lo que preferí sacar las opciones de un lugar remoto, y que posteriormente el usuario solo seleccione la suya.