**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

****

**ALUMNOS**

ALARCON GONZABAY LUIGGI STEVEN

ATIENCIA CALLE YSABEL YAMILETH

GOMEZ DE LA VERA KEVIN JOEL

RIVAS MORALES MIGUEL ANGEL

**PARALELO**

N°1

**MATERIA**

DISENO DE SOFTWARE

**TÉRMINO II 2018**

**Descripción del Sistema**

La plataforma permite que los estudiantes vendan sus productos y les permita receptar el pago mediante la plataforma.

Es de suma importancia una clase de conexiónSQL para el acceso a los datos de los usuarios, calificaciones, categorías, productos, etc. En el sistema existen dos tipos de clase heredadas de usuario, las cuales son vendedor y administrador. Debido a que vendedor puede realizar lo mismo que comprador, la clase vendedor heredará de comprador.

Las clases ventas y vendedor se encontrarán implementando el patrón de diseño observer para la implementación de la notificación al realizar el pago del producto, asimismo, producto ha sido implementado como una clase en el sistema. Debido a que existen diferentes formas de realizar los pagos, se usa la estrategia strategy en donde se usa la interfaz estrategiaPago la cual es implementada por estrategiaPago y estrategiaAppMovil.

Calificación ha sido implementado como clase para ser atributo de producto y vendedor, debido a que se deberá realizar búsquedas por productos, además, en clase categoría se encontrarán los productos que tengan una relación que asimismo servirá para búsquedas filtradas.

Tanto como para controladores y vistas existirá una clase para cada usuario los cuales se encargarán de mostrar y realizar los procesos que requieran los usuarios.

**Justificación de los patrones de diseño**

Los patrones de diseño ayudan a refinar los componentes o subsistemas de un sistema de información o las relaciones entre ellos. Además, resuelven un problema general de diseño, de manera eficiente y elegante, con un contexto particular.

En este caso, se ha logrado identificar los siguientes patrones de diseño.

*Patrón de diseño Singleton:*

El propósito de este patrón de diseño es lograr proporcionar un punto de acceso global a una variable de una clase que solo posee una instancia. Debido a que este patrón se aplica cuando debe existir un ejemplar de una clase y debe ser accesible a los clientes, se ha usado este patrón para la clase que logra la conexión a la base de datos, puesto que desde esta clase se logra acceder a los datos como los productos, usuarios, categorías, entre otros.

*Patrón de diseño Factory Method:*

Este patrón de diseño define la interfaz de creación de un cierto tipo de objeto, logrando así de que las subclases decidan los objetos que ella crea. Es usada cuando una clase no puede anticipar el tipo de objeto que debe crear. Debido a que no se puede anticipar la creación de una clase Vendedor o Comprador, se tiene una interfaz que delega su creación.

*Patrón de diseño Strategy:*

Strategy facilita la implementación de diferentes estrategias usando una clase común, con esto se ejecutará la estrategia concreta para la situación respectiva. Este patrón debe ser usado cuando se proporciona diferentes alternativas para realizar algo, como en este caso existe, por el momento, dos alternativas para realizar los pagos, se hace uso de este patrón de diseño.

*Patrón de diseño Observer:*

Observer permite reaccionar a ciertos eventos de clases que tienen como nombre “observadores”. Es implementado para monitorear estados de otros objetos para detectar algún tipo de evento. Debido a que es necesario notificar a los vendedores cuando un producto es pagado por algún comprador, es necesario usar este patrón para notificar a los vendedores para que así sepan que deben entregar el producto.