Principios de diseño Usados

Open/Closed Principle: Se usa este principio ya que se van expandiendo las clases, como por ejemplo de las clases de bicicletas se podrían ofertar más tipos de bicicletas y cada una en una clase diferente que hereda la clase principal de las bicicletas.

Single Responsibility Principle: Cada clase cumple su función única en el proyecto, así como hay una clase para login, también para registrarse, y a su vez para hacer la transacción de comprar o vender, y ver todos los usuarios en una tabla.

Liskov Substitution: Se utiliza liskov para crear la bicicleta que parte de la clase madre, cumpliendo que también se puedan usar estos con normalidad, en nuestro proyecto se usa para asignar las partes de la bicicleta personalizada.

Interface Segregation: Se usa ya que se van creando diferentes páginas a medida que el usuario avanza en la compra, y así tener varias interfaces.

Principio de Boy Scout: El proyecto se refactorizar con la idea de hacerlo lo más ordenado posible, cumpliendo este principio

Principio DRY: Se eliminaron funciones y métodos del proyecto que no fueran relevantes así como algunos que no fueran necesarios volver a crear si ya se tenían funcionalidades que servían para ese propósito, respetando el principio de responsabilidad única.

Principio YAGNI: Se eliminaron funcionalidades que estaban demás, solo se usaron las que se consideraban que mostraban un buen entendimiento de la asignatura, pero que también permitieran la coherencia del proyecto.

Principio KISS: Intentamos hacer el código lo más sencillo de entender, para que no haya confusiones con los nombres de las variables, así como el desarrollo del código.

Alta Cohesión y Bajo Acoplamiento: Se hace posible con este código que todos los módulos trabajen de manera lógica, pero cada uno tenga una funcionalidad distinta, y que no se combinen de manera que proporcionen un bajo acoplamiento.

Separation of Concerns: Cada módulo tiene una funcionalidad diferente a los cuales ciertos módulos no están conectados a otros módulos que por lógica y diseño no guardan relación, así como los servlets que a pesar de estar conectados, guardan cierta distancia lógica.