Actividad Módulo 3 - Patrones de Estructura de Software
Pre- especialización 2024

Integrantes del equipo:

Felix Gerardo Granadino Rauda
Mario Augusto Lue Morales
Fredy Alberto Benitez Gomez
Wilber Denilson Lopez Perez

Indicaciones generales:

1. La actividad va a ser desarrollada en equipos estructurados por afinidad con un máximo de 4 personas.
2. La fecha de entrega establecida para la actividad no tendrá prórroga.
3. Debe desarrollar cada uno de los ejercicios, mostrando un diseño de clase donde se vea reflejado el patrón de estructura de software mencionado en cada ítem.

https://creately.com/es/diagram-type/diagrama-clases/

4. No es necesario la codificación sino únicamente la representación gráfica de alto nivel de las clases involucradas de acuerdo al problema.

Criterios de evaluación de la actividad:

Criterio	Porcentaje
Puntualidad	20%
Calidad de la información presentada	50%
Seguimiento de las indicaciones	10%
Orden y Claridad en lo trabajado	30%
Total	100%

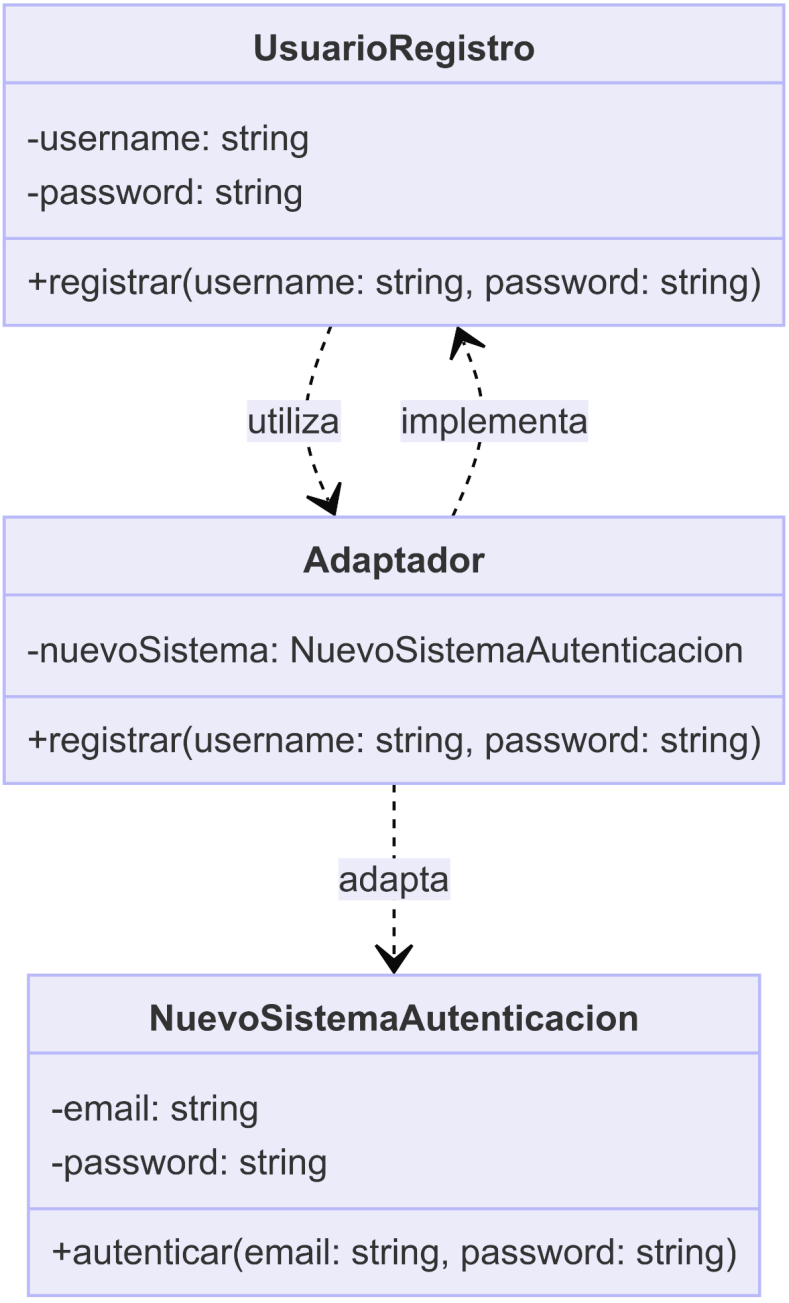
Criterios de evaluación del Módulo 3:

Criterio	Porcentaje
Asistencia sesión Meet	20%
Participación en chat o audio	20%
Actividad de Módulo	50%
Participación en actividades asincrónicas	10%
Total	100%

Adaptador (Adapter Pattern)

Problema: Tienes un sistema de registro de usuarios que utiliza una interfaz antigua que espera username y password, pero deseas integrar un nuevo sistema de autenticación que solo acepta email y password.

Tarea: Implementa un adaptador que permita al nuevo sistema de autenticación funcionar con la interfaz antigua de registro de usuarios.

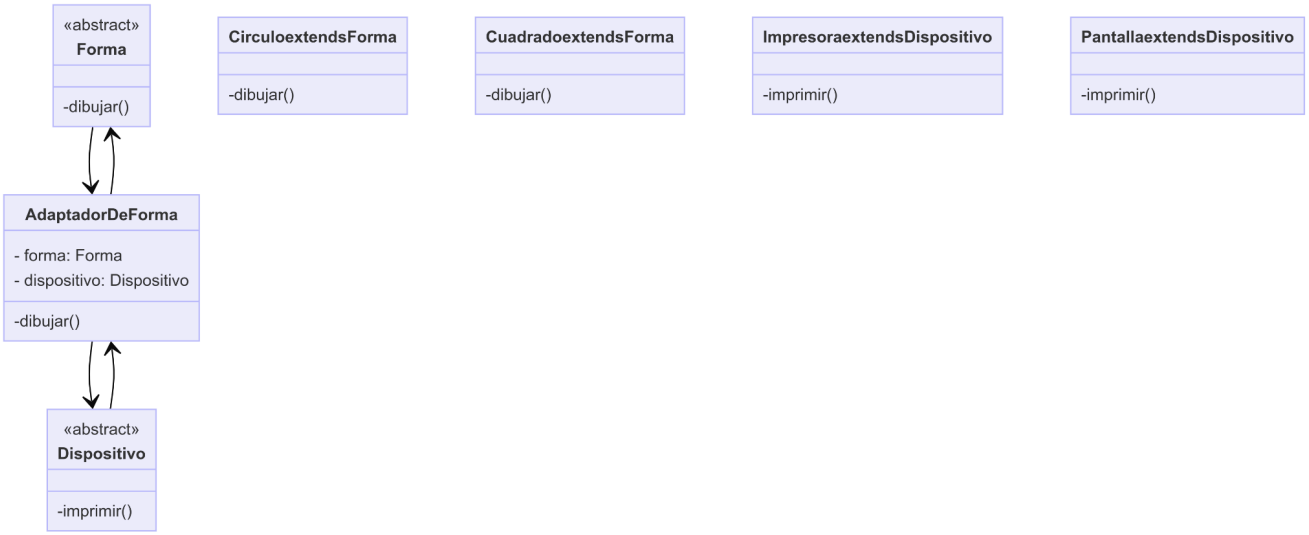


A

Puente (Bridge Pattern)

Problema: Estás construyendo un sistema de dibujo que debe funcionar con diferentes tipos de dispositivos de salida, como impresoras y pantallas. Deseas asegurarte de que puedas cambiar y agregar nuevos dispositivos sin afectar el código existente.

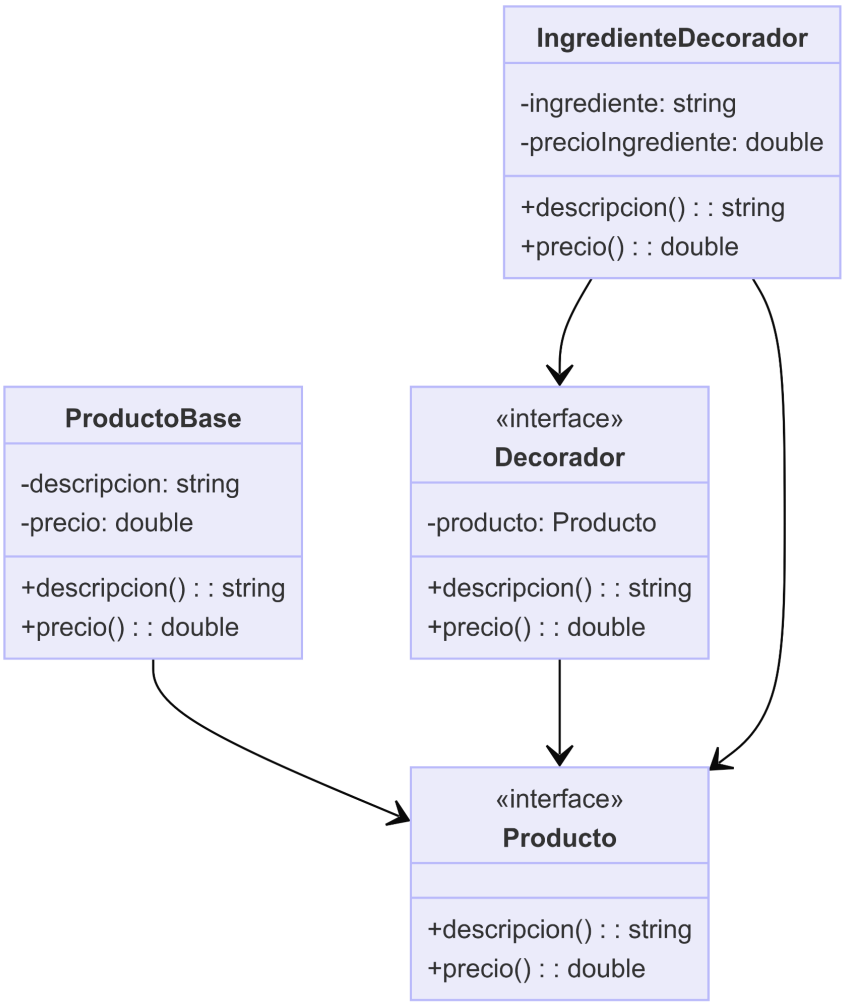
Tarea: Aplica el patrón Puente para separar las abstracciones (formas a dibujar) de las implementaciones (dispositivos de salida) y crea un sistema que funcione de manera flexible con diferentes dispositivos.



Decorador (Decorator Pattern)

Problema: Estás creando una aplicación de pedidos en línea que permite a los usuarios agregar ingredientes adicionales a sus productos. Deseas que el sistema sea flexible y permita que los usuarios agreguen varios ingredientes sin complicar la estructura del código base.

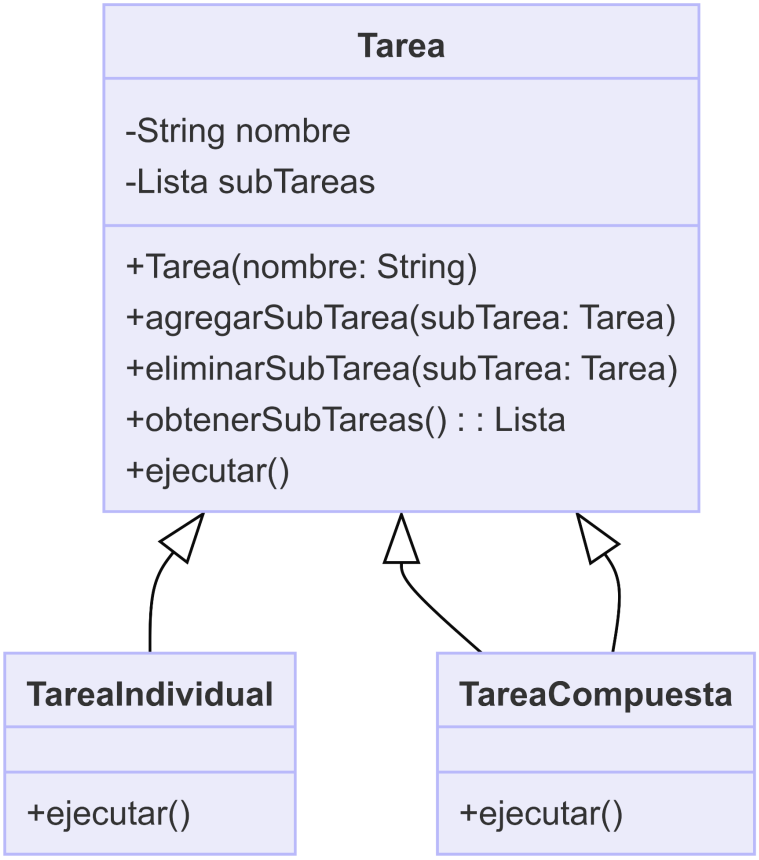
Tarea: Utiliza el patrón Decorador para permitir que los ingredientes adicionales se agreguen dinámicamente a los productos sin modificar su estructura principal.



Compuesto (Composite Pattern)

Problema: Estás construyendo un sistema de creación y gestión de tareas. Las tareas pueden ser individuales o compuestas por varias sub-tareas. Deseas tratar tanto las tareas individuales como las compuestas de manera uniforme en todo el sistema.

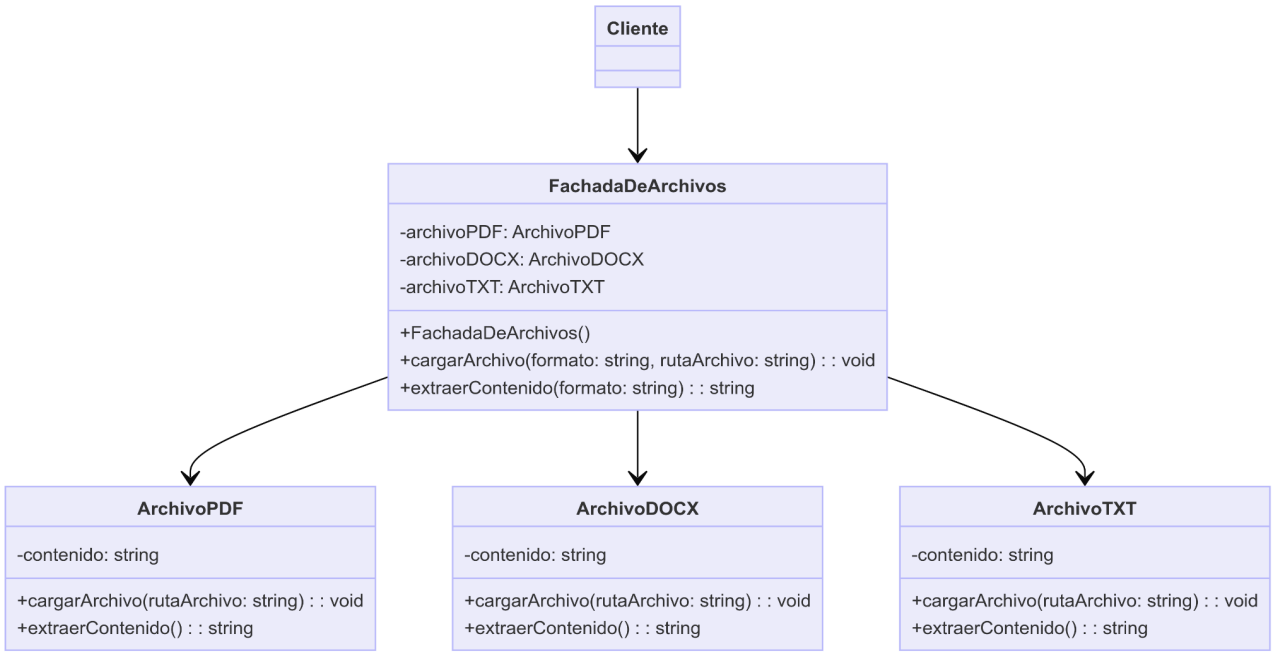
Tarea: Aplica el patrón Compuesto para permitir que las tareas individuales y las tareas compuestas se manejen de manera coherente, lo que facilita la gestión de la jerarquía de tareas.



Fachada (Facade Pattern)

Problema: Estás desarrollando una biblioteca para procesar y analizar archivos de diferentes formatos (PDF, DOCX, TXT). Deseas proporcionar a los usuarios una interfaz simple que oculte la complejidad de la manipulación de diferentes formatos.

Tarea: Implementa una fachada que ofrezca métodos simples para cargar y extraer contenido de archivos de diferentes formatos, ocultando los detalles de implementación.



Flyweight (Flyweight Pattern)

Problema: Estás desarrollando una aplicación de edición de imágenes que debe manejar múltiples objetos gráficos, como líneas, círculos y rectángulos. Deseas optimizar el uso de memoria para objetos que tienen propiedades similares.

Tarea: Aplica el patrón Flyweight para compartir eficientemente objetos gráficos que tienen propiedades similares, reduciendo la memoria utilizada por la aplicación.