Investigación:

1. Mencione al menos 4 ventajas de adoptar el patrón MVC en su desarrollo.

Se tienen muchas ventajas como:

* La implementación se realiza de forma modular.
* Sus vistas muestran información actualizada siempre. El programador no debe preocuparse de solicitar que las vistas se actualicen, ya que este proceso es realizado automáticamente por el modelo de la aplicación.
* Cualquier modificación que afecte al dominio, como aumentar métodos o datos contenidos, implica una modificación sólo en el modelo y las interfaces del mismo con las vistas, no todo el mecanismo de comunicación y de actualización entre modelos.
* Las modificaciones a las vistas no afectan al modelo de dominio, simplemente se modifica la representación de la información, no su tratamiento.
* MVC esta demostrando ser un patrón de diseño bien elaborado pues las aplicaciones que lo implementan presentan una extensibilidad y una mantenibilidad únicas comparadas con otras aplicaciones basadas en otros patrones.

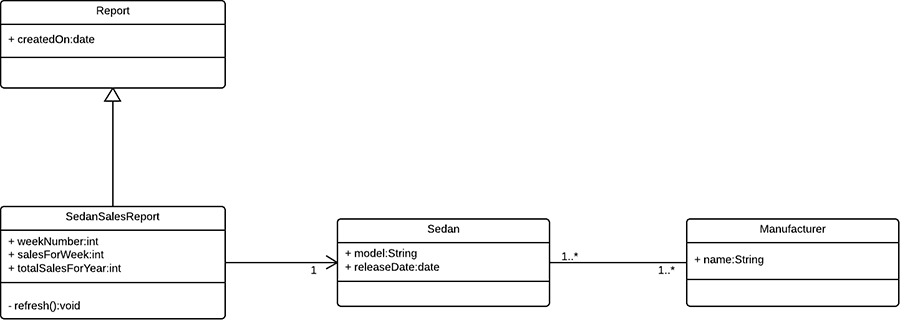
1. Mencione al menos 4 beneficios de implementar el uso del diagrama de clases en un proceso de desarrollo.

Los cuatro diagramas estructurales de UML existen para visualizar, especificar, construir y documentar los aspectos estáticos del sistema. Están organizados sobre grupos de objetos que se encontrarán cuando se esté modelado un sistema.

**Diagrama de clases**:

Los diagramas de clases representan las estructuras estáticas de un sistema, incluidas sus clases, atributos, operaciones y objetos. Un diagrama de clases puede mostrar datos computacionales u organizacionales en la forma de clases de implementación y clases lógicas, respectivamente. Puede haber superposición entre estos dos grupos.

1. Las clases se representan con una forma rectangular dividida en tercios. La sección superior muestra el nombre de la clase, mientras que la sección central contiene los atributos de la clase. La sección inferior muestra las operaciones de la clase (también conocidas como métodos).
2. Agrega formas de clases a tu diagrama de clases para modelar la relación entre esos objetos. Además, podría ser necesario que agregues subclases.
3. Usa líneas para representar asociación, traspaso, multiplicidad y otras relaciones entre clases y subclases. Tu estilo de notación preferido informará la notación de estas líneas.



Ejemplo de diagrama UML de interfaz de clases.

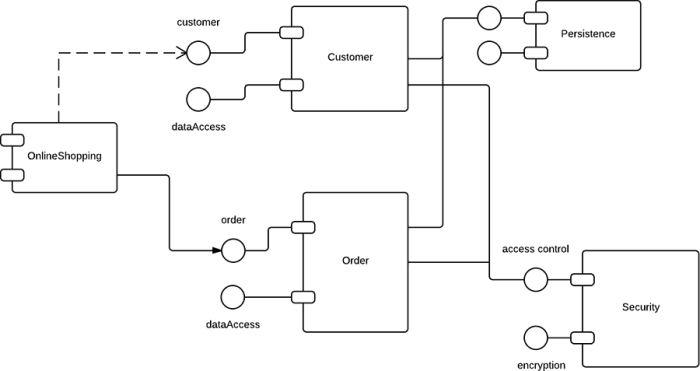
**Diagrama de objetos**:

A menudo, este diagrama se usa como una forma de comprobar la revisión de un diagrama de clases para fines de precisión. En otras palabras, ¿funcionará en la práctica? Muestra los objetos de un sistema y sus relaciones, y ofrece una mejor visión de los potenciales defectos de diseño que necesitan reparación.

**Diagrama de componentes**:

Los diagramas de componentes muestran cómo se combinan los componentes para formar componentes más grandes o sistemas de software. Estos diagramas están diseñados para modelar las dependencias de cada componente en el sistema. Un componente es algo necesario para ejecutar una función de estereotipo. Un estereotipo de componente puede constar de ejecutables, documentos, tablas de bases de datos, archivos o archivos de bibliotecas.

1. Representa un componente con una forma rectangular. Debe tener dos rectángulos pequeños en un lado o mostrar un icono con esa forma.
2. Agrega líneas entre formas de componentes para representar las relaciones pertinentes.



**Diagramas de implementación:**

Un diagrama de implementación modela la implementación física y la estructura de los componentes de hardware. Los diagramas de implementación muestran dónde y cómo operarán los componentes de un sistema en conjunto con los demás.

1. Al trazar un diagrama de implementación, usa la misma notación que usas para un diagrama de componentes.
2. Usa un cubo 3D para modelar un nodo (lo cual representa una máquina física o máquina virtual).
3. Etiqueta el nodo con el mismo estilo que se usa para los diagramas de secuencia. Agrega otros nodos según sea necesario, luego conéctalos con líneas.

