**1. Indique las capas en las que se dividen la arquitectura limpia.**

* Entidades
* Casos de uso
* Adaptadores
* Frameworks y drivers

**2. Clasifique los siguientes componentes de acuerdo a las capas de clean arquitectura:**

Frameworks y controladores

* Web
* Databases
* UI
* Infraestructura
* API's
* Dispositivos

Adaptadores de interface

* Gateways

Regla de negocio de aplicación

* Casos de Uso

Reglas de negocio empresariales

* Entidades

**3. Describa el principio de abstracción definido en arquitecturas limpias.**

Este principio nos dice que si un componente de nuestro sistema va a cambiar poco ya que es difícil modificarlo, debe estar compuesto mayoritariamente por interfaces y clases abstractas. De esta manera el componente será fácilmente extensible, y no afectará tanto al resto de la arquitectura.

**4. Es posible que una capa haga referencia a una capa interna de la arquitectura?**

violan el principio de inversión de dependencia porque la capa de dominio depende de la capa de acceso a datos, y el principio de inversión de dependencia dice: Las abstracciones no deben depender de los detalles los detalles deben depender de las abstracciones. Como ya sabemos la capa de dominio o negocio es la abstracción de algún dominio del mundo real y no debe depender de los detalles como la capa de datos, aplicación o presentación.

**5. Cual es la definición de API?**

API o *Application Programming Interface*, que en español quiere decir Interfaz de Programación de Aplicaciones, es un conjunto de funciones y procedimientos que permite integrar sistemas, permitiendo que sus funcionalidades puedan ser reutilizadas por otras aplicaciones o software.

**6. En sus propias palabras especifique acoplamiento y cohesión he indique cual es la mejor configuración que se debe mantener para que un sistema conserve los principios de arquitecturas limpias.**

* La cohesión es la medida en la que un componente o clase realiza únicamente la tarea para la cual fue diseñada (Una clase debe de hacer lo que respecta a su entidad, y no hacer acciones que involucren a otra clase o entidad).
* El acoplamiento es la medida que un componente o clase dependen de otro, generando cambios externos o alterando la funcionalidad del mismo (Cuando modificamos los atributos de una clase, se tienen que modificar los atributos de otra clase.)

En resumen podemos afirmar que en la definición de un buen diseño de software se debe tener una ALTA COHESIÓN y un BAJO ACOPLAMIENTO.

**7. Mencione y describa las partes de los casos de uso.**

Actor

* Representa a una persona o grupo de personas que desempeñan un papel en la interacción con el software.
* También puede ser “cualquier elemento” externo que interactúe con el software para lograr determinados objetivos, como por ejemplo otros sistemas, productos de software y equipamiento.

Caso de uso

* Representa [una funcionalidad](http://www.pmoinformatica.com/2018/05/que-es-requerimiento-funcional.html) (Conocida en inglés como Feature) que cumple uno o varios requisitos. Al colocarles nombre es recomendable usar verbos infinitivos con un complemento. Por ejemplo “Registrar nuevo cliente” en lugar de “Registro de nuevo cliente”.

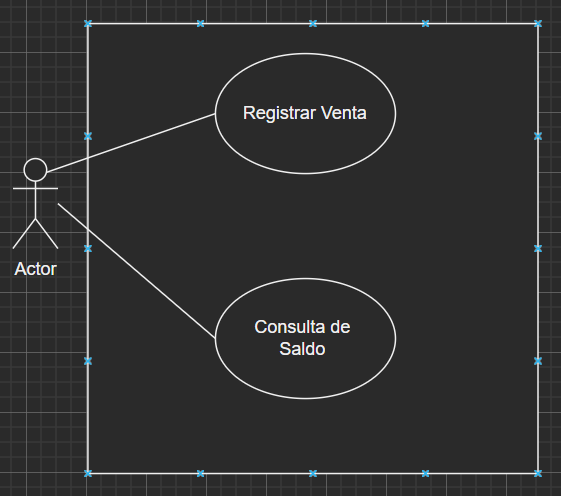
Relación

* La interacción entre dos casos de uso o de un actor con un caso de uso se representa por medio de una relación, utilizando una línea o una flecha.

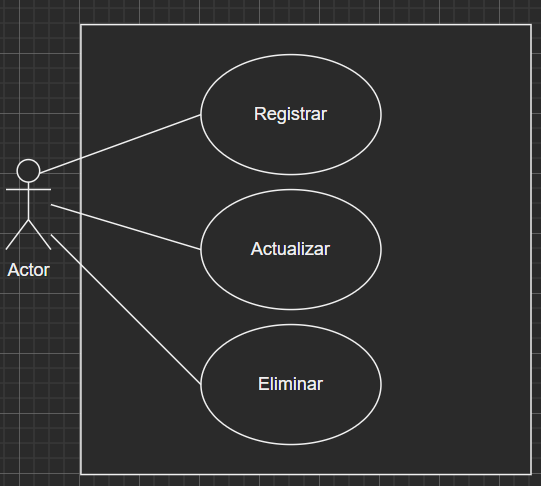
**8. Diagramación de casos de uso (tiempo 40 minutos):**

Una empresa comercializadora, desea invertir en el desarrollo de un sistema orientado al control de las ventas de sus productos, los requerimientos que el sistema debe satisfacer son:

A. Permitir que los Vendedores registren las ventas, para ello deberán consultar el saldo de los productos.



B. Permitir al Jefe de Recursos Humanos administrar al personal de ventas, registrando, actualizando y eliminando a los diferentes vendedores que ingresen o salgan de la empresa.



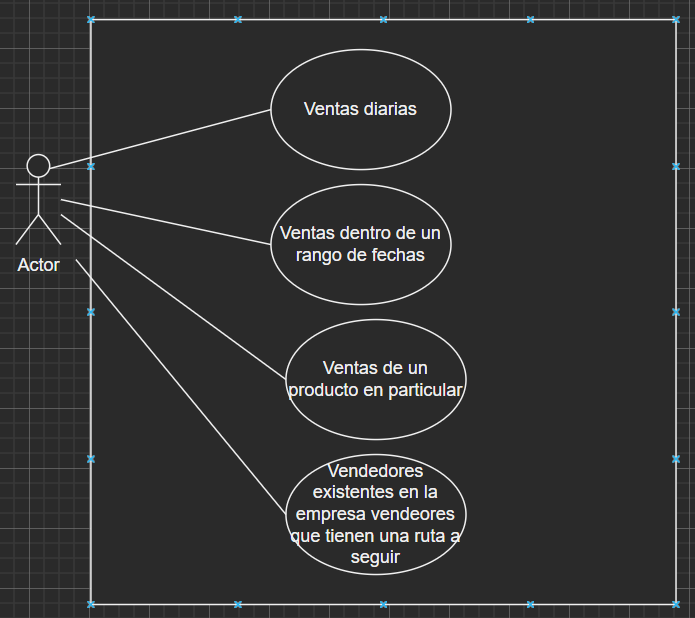
C. El Gerente de Ventas podrá consultar los siguientes reportes:

Ventas diarias.

Ventas dentro de un rango de fechas.

Ventas de un producto en particular.

Vendedores existentes en la empresa Vendedores que tienen una ruta a seguir.



9. Teniendo en cuenta el ejemplo anterior y luego de definir los casos de uso especifique que componentes debería tener y en a que capa de arquitecturas limpias pertenece.

Frameworks y controladores 🡪 Databases🡪 para consultar

Adaptadores de interface 🡪 Gateways 🡪 conexión de red con el sistema.

Regla de negocio de aplicación 🡪 Casos de Uso 🡪 Los casos de uso, solo definen como se comporta nuestro sistema, definiendo los datos de entrada necesarios, y cuál será su salida.

Reglas de negocio empresariales 🡪 Entidades 🡪 Diferentes procesos de casos de uso apunta a la regla principal para la primera consulta de saldo y vender. Para el segundo caso caso poder realizar operaciones básicas de base de datos y por ultimo 3 casos todo el tema de reportes reportes requerido por el usuario.

