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

Las capas que componen la arquitectura limpia son:

* Capa de Dominio: Compone las reglas de negocio general para todos los sistemas
* Capa de Aplicación: Reglas de negocio propias de la aplicación, llamados casos de uso
* Capa de Adaptadores: Son las interfaces que conectan la infraestructura con la aplicación. Pueden ser APIs, Gateways, etc.
* Capa de Infraestructura: Componentes externos o de infraestructura como frameworks, drivers, interfaz de usuario, bases de datos, etc.

2. Clasifique los siguientes componentes de acuerdo con las capas de clean architecture:

Entidades -> Capa de Aplicación o capa de Dominio

Infraestructura -> Capa de Infraestructura

API's -> Capa de Adaptadores

Gateways -> Capa de Adaptadores

UI -> Capa de Infraestructura

Casos de Uso -> Capa de Aplicación

Web -> Capa de Infraestructura

Databases -> Capa de Infraestructura

Dispositivos -> Capa de Infraestructura

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

Las reglas de negocio deben definirse únicamente en las capas de Dominio y de Aplicación. Se debe abstraer muy bien el comportamiento para que en las capas de Adaptadores e Infraestructura no se implementen reglas de negocio.

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

Sí, de lo contrario no se estaría cumpliendo con el principio de la arquitectura limpia de las dependencias no cíclicas, dado que la relación entre capas debe ser de una más externa a una más interna y nunca de adentro hacia afuera.

5. Cuál es la definición de API?

Es una interface que permite la conexión entre dos sistemas independiente de su tecnología y que se implementa a través de servicios.

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.

Acoplamiento hace referencia a la dependencia que existe entre componentes, lo ideal es que una clase no tenga una dependencia directa con otra clase. Además, existe acoplamiento si al hacer un ajuste en un módulo se deben modificar otros.

Cohesión hace referencia a la relación que existe entre componentes. Lo que se busca es que los objetos de dominio o clases relacionadas se agrupen, pero respetando el dominio al cual pertenecen, es decir, que estas no afecten a otros componentes.

En un proyecto siempre se debe buscar bajo acoplamiento y alta cohesión.

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

- Actor: Usuario/Sistema que va a ejecutar determinada acción

- Caso de Uso: Funcionalidad/Acción a realizar sobre el sistema

- Relación: Relación entre los actores y los casos de uso y entre los diferentes casos de uso

- Límites del Sistema: Agrupar casos de uso dentro de un mismo sistema

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.

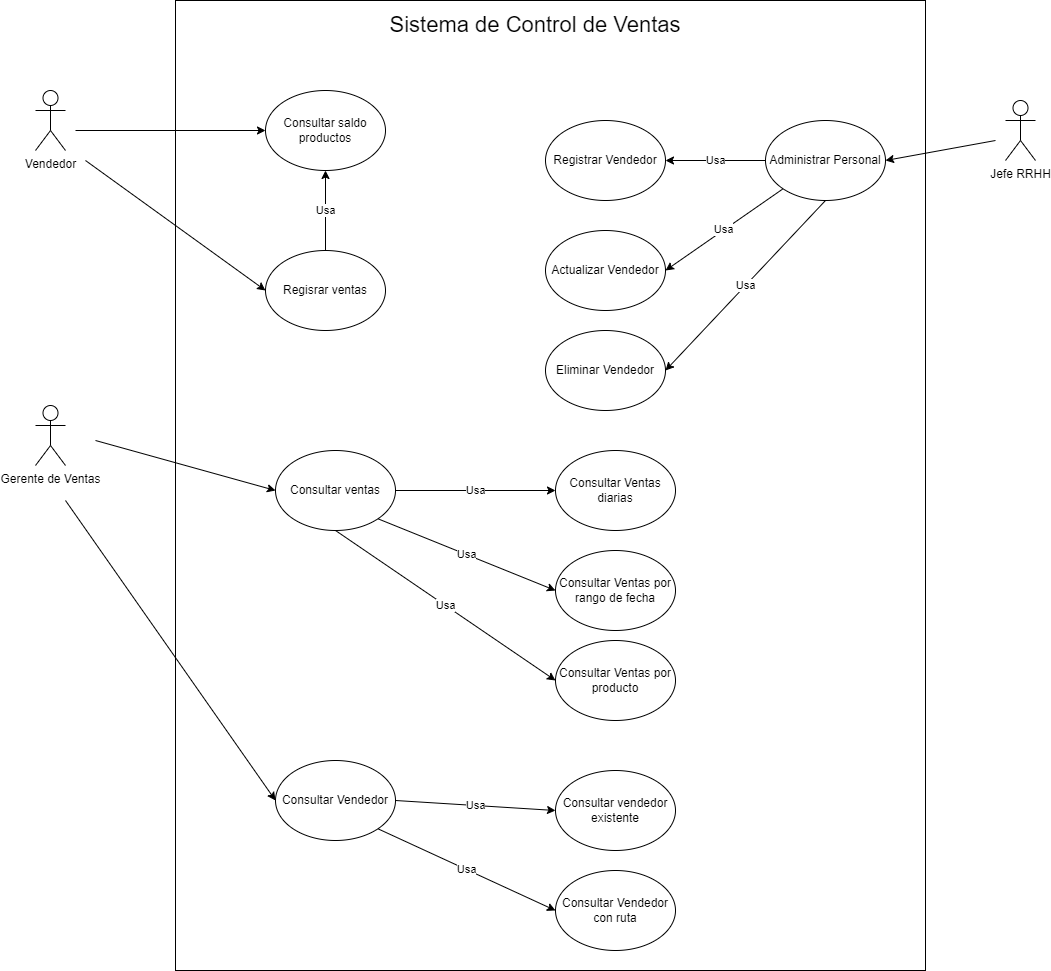
C. El Gerente de Ventas podrá consultar los siguientes reportes:

Ventas diarias.

Ventas dentro de un rango de fechas.

Ventas de un producto en particular.

Vendedores existentes en la empresa o 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 a que capa de arquitecturas limpias pertenece.

