{ADD-001.2}

* Nombre Decisión: [Estilo por Capas]
* Estatus: [Propuesto]
* Fecha: [2022-11-08]

**Requisitos**

\*[RF-02] \*[RF-03] \*[RF-09]

**Descripción**

{Consta en dividir la aplicación en capas, con la intención de que cada capa tenga un rol muy definido, como podría ser, una capa de presentación (UI), una capa de reglas de negocio (servicios) y una capa de acceso a datos (DAO), sin embargo, este estilo arquitectónico no define cuantas capas debe de tener la aplicación, sino más bien, se centra en la separación de la aplicación en capas (Aplica el principio Separación de preocupaciones (SoC)).

**Argumentación**

{Ya que son los sensores los que se comunican con el cockpit y es este el que permite visualizar las analíticas, podemos separar cada nivel por capas y así tener la responsabilidad de cada capa por separado}

**Consecuencias Positivas**

* Separación de responsabilidades: permite que cada capa se encargue estrictamente de su trabajo y no se vea afectada por las responsabilidades de otras capas de otro nivel
* Fácil de desarrollar
* Fácil de probar: al estar separado por capas, el testing se puede hacer sin problemas en cada capa de manera individual
* Fácil mantenimiento: ya que tenemos las capas separadas, encontrar un fallo y corregirlo resulta en una tarea sencilla
* Seguridad contra ataques: ya que las capas están separadas en niveles, al recibir un ataque en una capa, el resto no se ven afectadas

**Consecuencias Negativas**

* Velocidad: al estar separado en capas, la comunicación entre ellas no es del todo fluida y puede ralentizar ciertos procesos
* Baja escalabilidad
* Alta complejidad y dependencias al desplegar los componentes
* Fallos en cascada: si una de las capas falla, lo más probable es que el resto del sistema colapse