UNIVERSIDAD DE LAS FUERZAS ARMADAS-ESPE - MATRIZ

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN - DCCO-SS

CARRERA DE INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

ASIGNATURA : MET. DE DESARROLLO SOFTWARE

TEMA : P3_TALLER1

ESTUDIANTE : Lucio Jelen, Cahuatijo Noe, Cevallos Bryan, Bolaños Leopoldo

NIVEL-PARALELO - NRC: Tercero A – NRC : 29022

DOCENTE : Ing. Ruiz Jenny

FECHA DE ENTREGA : 09/01/2026.

SANGOLQUI – ECUADOR

Ejemplo:

Revisar	Qué debe cumplirse	Error común
Nombres	Clases en singular y con significado del dominio.	Nombres genéricos (Datos, Manager, Utils).
Multiplicidad	Cada asociación relevante tiene cardinalidad.	Omitir multiplicidades y perder reglas del negocio.
Cohesión	Responsabilidades claras y relacionadas.	Una clase concentra demasiadas funciones.
Acoplamiento	Dependencias mínimas y justificadas.	Asociaciones por conveniencia sin razón del dominio.
Trazabilidad	Métodos justificados por escenarios (secuencias).	Inventar métodos sin escenario real.

Tabla de casos de uso:

Criterio	**Qué debe cumplirse**	**Estado**	**Justificación del Diseño (Evidencia)**
Nombres	Clases en singular, verbos en infinitivo del dominio.	**Cumple**	Se renombraron los casos como: *Autenticar, Procesar venta, Actualizar stock*. Se eliminaron nombres genéricos como "Datos" o "Utils".
Multiplicidad	Cada asociación relevante tiene cardinalidad.	**Cumple**	Se incluyó la cardinalidad **1..1** en el Actor y **1..*** en los casos de uso para indicar que un usuario realiza múltiples acciones.
Cohesión	Responsabilidades claras y relacionadas.	**Cumple**	El caso *Actualizar stock (REQ004)* es altamente cohesivo: solo gestiona inventario. No se mezcla con facturación ni con datos de clientes.
Acoplamiento	Dependencias mínimas y justificadas.	**Cumple**	Se usó <<include>> solo para pasos obligatorios (ej. Venta -> Ticket) y <<extend>> para opcionales, evitando conexiones innecesarias.
Trazabilidad	Métodos justificados por los escenarios.	**Cumple**	Existe una relación directa 1 a 1 entre los óvalos del diagrama

Criterio	Qué debe cumplirse	Estado	Justificación del Diseño (Evidencia)
			y los requisitos REQ001 al REQ010 de la matriz de Historias de Usuario.

Tabla de Clases:

Revisar	Que debe cumplirse	Errores Comunes
Cohesión	Cada clase/módulo debe tener una única responsabilidad clara.	Que los módulos de la "Capa Lógica" terminen haciendo tareas de la GUI (como formatear texto para una tabla) o que la GUI ejecute consultas SQL directamente.
Trazabilidad de Métodos	Los métodos (ej: validar_login, obtener_historial) deben corresponder a interacciones de los casos de uso.	Agregar métodos "por si acaso" en la capa lógica que no tienen un botón o acción correspondiente en la capa de presentación
Multiplicidad	Las relaciones entre tablas de DB deben indicar cuántos objetos participan (ej: un Producto tiene un Proveedor).	En el diagrama de base de datos, omitir los indicadores 0..* o 1 en las flechas de producto_id o usuario_id, lo que genera confusión al implementar las llaves foráneas
Nombres	Usar sustantivos en singular y nombres con significado del dominio (ej: Movimiento, Proveedor).	Usar nombres genéricos o técnicos como DataModule o Table1 en lugar de nombres de negocio como LogicProducts.
Acoplamiento	Las dependencias deben ser mínimas y justificadas.	Que la clase Database dependa de clases de la GUI, creando un acoplamiento

		circular que impide reutilizar la lógica

Tabla de secuencia

Revisar	Que debe cumplirse	Errores Comunes
Controlador	Debe existir un objeto (ej. LogicAuth, LogicProducts) que orqueste el flujo entre la UI y la DB.	Que la interfaz (Frame) envíe datos directamente a la Database sin pasar por la capa lógica.
Mensaje a Métodos	Los mensajes deben ser verbos claros que representen métodos reales en el código.	Usar descripciones narrativas largas (ej. "el sistema guarda los datos") en lugar de nombres de métodos (ej. crear_producto()).
Fragmentos	Se deben modelar las validaciones y los caminos alternos (errores).	Modelar solo cuando una validación es correcta y omitir qué sucede cuando falla (ej. RUC duplicado o login incorrecto).
Activación	Las barras de activación deben mostrar el tiempo exacto que un objeto está procesando una tarea	Dejar barras de activación infinitas o que no coinciden con el inicio y fin del mensaje de retorno.
Trazabilidad	Cada mensaje enviado a un objeto debe estar definido como un método en su clase correspondiente.	Inventar métodos en la secuencia que no aparecen en el Diagrama de Clases o en la Base de Datos.