

Entregable 10 – Desarrollador.

Vega Ángeles Christopher

22140787

Ingeniería en Sistemas Computacionales, Instituto Tecnológico de México campus

Querétaro.

Ingeniería de Software.

6A

SCC1007

Villeda Maldonado Julio Alejandro

24 de febrero de 2026

“Entregable 10”

Caso de uso: Registrar paciente.

- Nivel de complejidad justificado: (Medio)

Algunos de los aspectos principales a manejar son: la concurrencia y la integridad referencial. Al trabajar con Oracle/PL/SQL, el registro implicara el uso de disparadores de auditoría y la validación de llaves primarias (CURP/ID) antes de agregar el registro de un nuevo paciente para evitar excepciones que lance la base de datos que ocasione romper el flujo normal de la UI

- Riesgos:

- Duplicación de registro: Crear expedientes para la misma persona por errores de captura. (ID O CURP)
- Inconsistencia de datos: Captura de formatos incorrectos. (Teléfonos, Correos, fecha de nacimiento)
- Inyección de código: Riesgos de seguridad si los campos no están saneados (SQL inyección)
- Latencia de Red: Si la validación del CURP se hace contra una API externa (RENAPO) y esta falla, el flujo se detiene.

- Dependencias.

- Módulo de Catálogos: Depende de tablas “Catalogo ” para las llaves foráneas (Estado, Municipio, sexo, Tipo de sangre, Derechohabienta, Alergias comunes, etc).

- Servicio de Autenticación: Generar un registrar del ID del Enfermero o Médico que realiza la transacción (trazabilidad).
- Observaciones técnicas.
 - Validaciones en backend y frontend para campos únicos (NSS, CURP)
 - Normalización de la base de datos: Correctas relaciones de las tablas.
 - Estandarizar el formato de captura de datos sensibles.
 - Persistencia: La operación debe ser automática; no debe de crear un expediente vacío si el registro de datos personales falla.
 - El uso del fragmento “loop” en el diseño debe controlarse con un máximo de intentos para evitar bloqueos en el hilo de ejecución de la interfaz.
- Impacto estructural.
 - Afecta directamente la tabla principal de” PACIENTES” y las tablas de auditoría. Se tiene que mantener un perfecto control en relación a los datos ingresados, ya que si existe algún error esto corromperá las tablas consecuente que dependan del paciente

Caso de uso: Dar de baja paciente

- Nivel de complejidad justificado: (Alto)

La complejidad no es el borrado, sino la realizar una baja lógica. En sistemas clínicos, no se debe aplicar un DELETE físico por cumplimiento legal y normativo.

- Riesgos:
 - Pérdida de historial clínico (Conservar por periodos de 5- 10 años según la normativa legal).
 - Datos huérfanos: Dejar registro de consultas, recetas o laboratorios apuntando a un paciente que ya no existe en el índice activo.
 - Accesos no autorizados: Un usuario sin privilegio administrativos pueda realizar esta acción.
 - Acciones accidentales: Que un médico realice una baja por error al no tener un paso de confirmación visualmente descriptivo.
 - Acceso Post-Baja: Intentar hacer una nueva consulta a un paciente cuyo estado es “baja”, lo que generaría error en la consulta del servidor.
- Dependencias.
 - Módulo de Citas: Debe existir un disparador o proceso que cancele automáticamente las citas pendientes al cambiar el estado del paciente a "Baja".
 - Repositorio Paciente: Requiere un método que actualice el estado específico del paciente.

- Observaciones técnicas.
 - Bitácora de auditoria (Trigger): Al confirmar la baja (paso 8) Registrar obligatoriamente quién realizó la baja, en qué fecha y bajo qué motivo, el estado anterior del paciente.
 - Borrado lógico: Utilizar una bandera de estado en lugar de eliminar la fila de la tabla. (Ejecutar un UPDATE en vez de un DELETE en estado del paciente.)
 - Indexación de búsqueda para consultas eficientes en campos como Nombre, id y CURP
- Impacto estructural.
 - Impacto en la Persistencia de datos: Todas las consultas de búsqueda (SELECT) en otros módulos deben de validar con un filtro, para evitar que pacientes dados de baja "contaminen" la búsqueda normal del sistema.