

# CASOS DE USO

## Introducción

Como parte del proceso de revisión y priorización de requisitos del sistema, se determinó que los casos de uso asociados al acceso, autenticación y seguridad constituirían elementos críticos para la integridad de la información y la protección de los actores involucrados.

Por ello, se priorizó la documentación detallada y ampliada de los casos de uso relacionados con el acceso al sistema, registro de usuarios, validación de credenciales, recuperación de contraseñas, cierre de sesión y acceso a los paneles operativos (Dashboard).

Los siguientes casos de uso contienen descripciones detalladas sobre: reglas de negocio, validaciones de entrada, formatos esperados, flujos alternos y excepcionales, mensajes de error, mecanismos técnicos sugeridos (encriptación, hashing, expiraciones, logging), datos que deben persistir, y consideraciones para auditoría y cumplimiento.

## RF-1: Gestionar Acceso al Sistema

### Descripción General (ampliada)

El sistema gestionará el acceso y la identidad de tres tipos de actores: **Cliente**, **Asistente** y **Gerente**. El ciclo de vida de acceso incluye: registro (cuando aplique), autenticación, autorización por rol, manejo y expiración de sesiones, recuperación de credenciales y revocación de accesos.

Requisitos clave y detalles de implementación recomendados:

- **Modelo de identificación:** El usuario se identifica por un atributo único, preferiblemente correo electrónico verificado (campo `email`). Internamente, se debe usar un `user_id` UUID para relaciones y trazabilidad.
- **Autenticación:** Contraseñas almacenadas con **hash adaptativo** (por ejemplo bcrypt con cost factor configurable o Argon2). Uso de sal por usuario. No almacenar contraseñas en texto plano.

- **Autorización basada en roles (RBAC):** Definir permisos por rol (Cliente, Asistente, Gerente). El acceso a endpoints y vistas debe validarse en backend mediante middleware que verifique token y permisos.
- **Gestión de sesiones:** Tokens JWT de corta duración (por ejemplo 15-60 min) más refresh tokens almacenados en base de datos con posibilidad de revocación. Alternativa: sesiones server-side con identificadores aleatorios inmutables.
- **Protecciones anti-abuso:** Límite de intentos de inicio (por IP y por cuenta), bloqueo temporal (por ejemplo 15-30 min) y notificación al correo del usuario tras múltiples intentos fallidos.
- **Recuperación de cuenta:** Código numérico de 6 dígitos o enlace único con token cifrado y expiración corta (por ejemplo 10 minutos). El token debe ser de un solo uso y registrarse en logs.
- **Registro y auditoría:** Registrar en logs auditablemente: intentos de inicio (fecha, IP, user\_id, resultado), cambios de contraseña, creación y eliminación de cuentas, concesión y revocación de roles.
- **Protección de datos:** Cerciorarse de cifrado en tránsito (TLS 1.2/1.3) y cifrado en reposo para datos sensibles (por ejemplo campos PII o respaldos).
- **Usabilidad y accesibilidad:** Formularios con validaciones en cliente y servidor, mensajes claros de error y ayudas contextuales. Soporte para lectores de pantalla y navegación por teclado.
- **Internacionalización:** Textos y mensajes parametrizables (i18n).

Código	Descripción
RNF-01	Seguridad de acceso: hashing de contraseñas, almacenamiento con sal, políticas de bloqueo y MFA opcional.
RNF-02	Disponibilidad: mecanismos de failover y alta disponibilidad para el módulo de autenticación.
RNF-03	Usabilidad: formularios guiados, validaciones asíncronas, mensajes locales.
RNF-04	Respaldo: cifrado y rotación de backups con política de retención definida.
RNF-09	Registro y monitoreo: logs con integridad y retención para auditoría.

## RF-1.1: Registrar Cliente

### Descripción (extensa)

Proceso por el cual una persona natural crea una cuenta de tipo **Cliente** en el sistema para acceder a servicios. El registro puede ser realizado por el propio cliente a través del formulario público o por un operador autorizado en casos especiales (por ejemplo creación asistida).

#### Datos de entrada obligatorios:

- **nombre\_completo**: texto (3–50 caracteres). Permitidos letras (incluyendo tildes), espacios y signos de composición (p. ej. “María del Carmen”). No se aceptan números en este campo.
- **email**: correo electrónico (unicidad obligatoria). Validar por expresión regular robusta y normalizar a minúsculas.
- **password**: contraseña con política de complejidad (mínimo 8 caracteres, al menos 1 mayúscula, 1 minúscula, 1 número y 1 carácter especial).
- **confirm\_password**: coincidencia exacta con **password**.

#### Validaciones servidor-cliente:

- Validación en cliente para feedback inmediato y validación en servidor como única fuente de verdad.
- Verificar que el correo no exista ya en la base de datos (consulta atómica para evitar race conditions).
- Forzar hashing con salt (bcrypt/Argon2) antes de persistir.
- Generar un **user\_id** UUID y marcar estado **pending\_verification** hasta confirmar email si la política exige verificación.

#### Mensajes y UX:

- Mensajes claros para cada error de validación (ej. “La contraseña debe tener al menos 8 caracteres y un carácter especial” — evitar mensajes que filtren si el correo existe o no para prevenir enumeración).
- Mostrar opción de visualizar contraseña (eye icon) y ayudas sobre requisitos.

#### Persistencia y datos a registrar:

- Tabla `users`: `user_id`, `email`, `password_hash`, `salt`, `full_name`, `role`, `status`, `created_at`, `created_by`, `email_verified_at` (nullable).
- Log de auditoría: evento “`user.register`” con `user_id`, `ip`, `user_agent`, `timestamp`, `source`.

#### Notificaciones:

- Envío de correo de confirmación con token firmado o código de 6 dígitos. Plantilla debe incluir un enlace que expire (ej. 24h para verificación inicial).

#### Políticas de seguridad adicionales:

- Rechazo de contraseñas del top N (listas de contraseñas comunes).
- ReCAPTCHA o mecanismo anti-bot para prevenir registros masivos.
- Registro de origen (IP, país) para análisis de riesgo.

La Figura 1 muestra el caso de uso del RF-1.1: Registrar Cliente.

Elemento	Descripción
<b>Actor</b>	Cliente (usuario no autenticado que accede al formulario público).
<b>Precondición</b>	Acceso al formulario de registro; el correo no debe estar registrado.
<b>Secuencia Normal</b>	<ol style="list-style-type: none"> <li>1. El actor selecciona “Registrar” en la pantalla pública.</li> <li>2. El sistema despliega formulario con campos: nombre, email, contraseña, confirmar contraseña, checkbox de aceptación de términos.</li> <li>3. El usuario ingresa datos y envía el formulario.</li> <li>4. El servidor valida a nivel sintáctico y semántico; verifica unicidad del correo de forma atómica; valida fortaleza de contraseña.</li> <li>5. Si las validaciones son correctas: se crea registro en estado <code>pending_verification</code> (o <code>active</code> si no se exige verificación), se genera hash de contraseña con salt y se persiste.</li> <li>6. El sistema envía correo de verificación (o notificación de bienvenida) y muestra pantalla de confirmación.</li> </ol>

<b>Secuencias</b>	<b>Alter-</b>	<p>A. Campos obligatorios incompletos: el sistema resalta campos y devuelve mensajes en línea.</p> <p>B. Correo ya registrado: devolver mensaje genérico (ej. “Si ya existe una cuenta asociada, puede recuperar su contraseña”) para evitar enumeración.</p> <p>C. Contraseña débil: mostrar lista de requisitos no cumplidos.</p>
<b>Postcondición</b>		Cuenta creada; usuario en estado <code>pending_verification</code> o <code>active</code> .
<b>Excepciones</b>		Error de persistencia (registro no completado): registrar incidencia, mostrar mensaje: “Error en el registro. Intente más tarde” y reintentar transaccionalmente.
<b>Comentarios</b>		Se recomienda enviar evento a sistema de monitorización (ej. Sentry/ELK) y generar alerta si hay picos inusuales de registros.

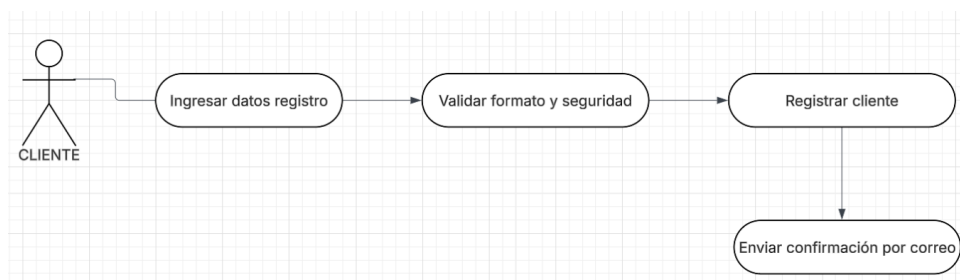


Figura 1: Diagrama del Caso de Uso – RF-1.1 Registrar Cliente. La figura muestra actores, formulario y flujo de verificación.

## RF-1.2: Registrar Nuevo Asistente

### Descripción (extensa)

Funcionalidad exclusiva para el rol **Gerente** (o un usuario con permiso administrativo) que posibilita crear cuentas de tipo **Asistente** con datos personales y una contraseña inicial temporal. Este proceso debe estar sujeto a controles adicionales para garantizar que solo usuarios autorizados puedan realizarlo.

#### Datos de entrada mínimos:

- `nombre_completo`, `email`, `id_identificacion` (opcional), `rol` (predefinido como `ASISTENTE`), `password_temporal`.

#### Reglas y validaciones:

- El gerente debe estar autenticado y su sesión activa. Verificar permiso ‘user:create<sub>a</sub>ssistant’.
- El correo debe pasar la verificación de formato y no existir.
- La contraseña temporal debe cumplir políticas básicas y marcarse con flag ‘must<sub>c</sub>hange<sub>p</sub>assword = true’.
- Registrar quién creó al asistente (‘created<sub>by</sub> = gerente<sub>u</sub>ser<sub>i</sub>d’)paratr azabilidad.

#### Notificaciones y activación:

- Enviar correo al asistente con instrucciones y enlace de activación que obliga al cambio de contraseña (token de un solo uso con expiración, p. ej. 48 horas).
- Registrar evento ‘assistant.created’ con metadatos: created<sub>by</sub>, timestamp, ip.

#### Seguridad operativa:

- Limitar el número de asistente que un gerente puede crear por día (política configurable).
- Mantener un log detallado y una vista en la UI del gerente con auditoría de acciones.

La Figura 2 muestra el caso de uso del RF-1.2: Registrar Nuevo Asistente.

Elemento	Descripción
<b>Actor</b>	Gerente (usuario con permiso administrativo).
<b>Precondición</b>	El gerente debe tener sesión activa y permiso explícito.
<b>Secuencia Normal</b>	<ol style="list-style-type: none"> <li>1. El gerente accede al módulo “Registrar Nuevo Asistente”.</li> <li>2. El sistema muestra formulario con campos requeridos y validaciones en línea.</li> <li>3. El gerente ingresa los datos y envía la solicitud.</li> <li>4. El servidor valida permisos del creador, corrige/normaliza entradas (p. ej. email a minúsculas), valida unicidad.</li> <li>5. El sistema crea el registro con contraseña temporal y flag ‘must<sub>c</sub>hange<sub>p</sub>assword’.</li> <li>6. Se envía correo de activación al asistente; se registra evento de auditoría.</li> </ol>
<b>Secuencias Alternas</b>	<ol style="list-style-type: none"> <li>A. Correo duplicado: informar y permitir introducir otro correo.</li> <li>B. Permisos insuficientes: mostrar mensaje “Acceso denegado”.</li> </ol>

<b>Postcondición</b>	Asistente creado y pendiente de activación; gerente tiene registro de la acción.
<b>Excepciones</b>	Fallo en servicio de mail o persistencia: el sistema debe encolar la notificación y dejar el registro en estado <code>created_pending_notification</code> .
<b>Comentarios</b>	Forzar el cambio de contraseña en el primer inicio y restringir contraseñas temporales en listas débiles.

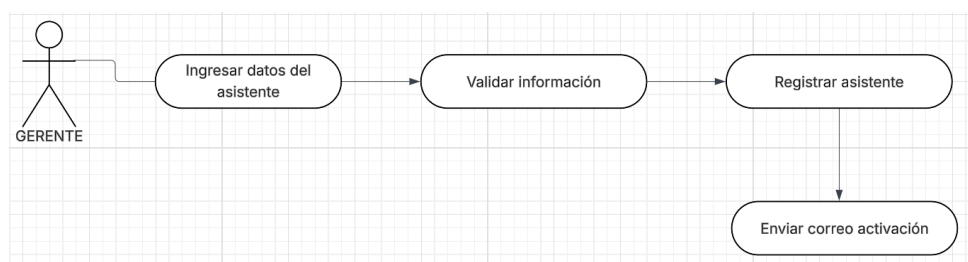


Figura 2: Diagrama del Caso de Uso – RF-1.2 Registrar Nuevo Asistente. La figura ilustra el flujo desde el gerente hasta la activación del asistente.

## RF-1.3: Iniciar Sesión

### Descripción (extensa)

Caso de uso que autentica a un usuario registrado y lo autoriza a acceder a las funcionalidades propias de su rol. Debe incluir protección contra ataques de fuerza bruta y mecanismo de registro de eventos de seguridad.

#### Flujo técnico recomendado:

- Usuario envía `email` y `password` al endpoint de autenticación (POST `/api/auth/login`).
- Backend valida formato del email; busca registro por email; compara hash de contraseña con `bcrypt/Argon2`.
- Si credenciales válidas: generar token de acceso y refresh token; persistir refresh token con TTL; registrar evento de login exitoso (timestamp, IP, `user_agent`, location if available).
- Si credenciales inválidas: incrementar contador de fallos; si supera umbral, bloquear cuenta temporalmente y notificar al propietario del correo.

#### Seguridad adicional:

- Soporte opcional de MFA (TOTP o SMS) para roles privilegiados (Gerente).

- Protección CSRF para sesiones basadas en cookies; usar SameSite=strict y cookies seguras.
- Limitar tamaño de payloads; sanitizar entradas para prevenir inyección.
- Uso de HTTPS obligatorio.

La Figura 3 muestra el caso de uso del RF-1.3: Iniciar Sesión.

Elemento	Descripción
<b>Actor</b>	Cliente, Asistente, Gerente.
<b>Precondición</b>	Usuario registrado y no bloqueado; servicio de autenticación operativo.
<b>Secuencia Normal</b>	<ol style="list-style-type: none"> <li>1. El usuario accede a la interfaz de “Iniciar Sesión”.</li> <li>2. Ingresa email y contraseña.</li> <li>3. El sistema valida sintaxis y existencia del correo.</li> <li>4. El sistema compara hash de contraseña.</li> <li>5. Si es exitoso, crea sesión/token y redirige al Dashboard correspondiente según rol.</li> <li>6. Registra evento de acceso exitoso en logs de auditoría.</li> </ol>
<b>Secuencia Alterna</b>	<ol style="list-style-type: none"> <li>A. Contraseña incorrecta: incrementar contador de fallos; mostrar mensaje genérico “Credenciales incorrectas”.</li> <li>B. Usuario bloqueado por intentos: mostrar mensaje “Cuenta temporalmente bloqueada” y enviar correo informativo.</li> <li>C. Requerir MFA: si usuario tiene MFA habilitado, solicitar código TOTP antes de emitir tokens.</li> </ol>
<b>Postcondición</b>	Usuario autenticado; sesión válida con tokens y registros de auditoría.
<b>Excepciones</b>	Caída del servicio de autorización, problemas de base de datos: devolver error 503 y registrar incidente.
<b>Comentarios</b>	Registrar metadatos: <code>user_id</code> , <code>ip</code> , <code>user_agent</code> , <code>device_fingerprint</code> , <code>timestamp</code> , <code>outcome</code> . Mantener política de rotación de tokens y revocación por logout o cambio de credenciales.

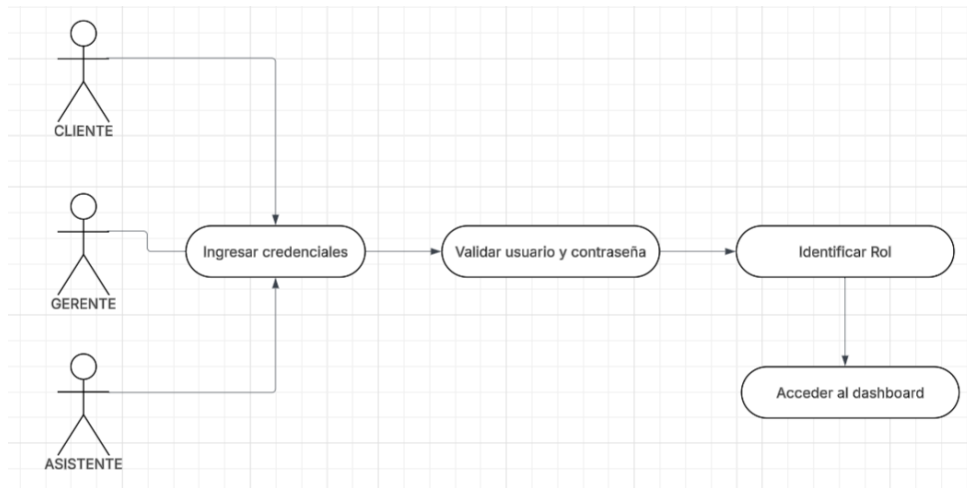


Figura 3: Diagrama del Caso de Uso – RF-1.3 Iniciar Sesión. La figura muestra autenticación, control de intentos y emisión de tokens.

## RF-1.4: Recuperar Contraseña

### Descripción (extensa)

Flujo seguro que permite a un usuario recuperar el acceso a su cuenta cuando ha olvidado la contraseña. Debe evitar filtraciones de información y ser resistente a ataques de enumeración y replay.

#### Opciones soportadas:

- Código numérico de 6 dígitos enviado por correo (o SMS si se habilita).
- Enlace de recuperación con token firmado y expiración (ej. 10 minutos).

#### Requisitos de seguridad:

- Tokens de un solo uso; marcar como consumido tras uso.
- Limitar solicitudes por hora por cuenta/IP; bloqueo temporal si se detectan abusos.
- Registrar cada solicitud con `request_id`, `user_id`, `ip`, `timestamp`.
- Confirmación por email para notificar cambios de contraseña.

#### Proceso:

1. Usuario solicita recuperación en “¿Olvidaste tu contraseña?” indicando su email.
2. Sistema verifica existencia del email sin revelar si existe (mostrar mensaje genérico para el usuario: “Si existe una cuenta asociada, recibirá instrucciones”).

3. Generar token único y enviar por correo o código numérico. Registrar evento.
4. Usuario usa token o código para acceder a formulario de cambio. Validar expiración y consumo único.
5. Usuario establece nueva contraseña que cumpla política de seguridad. Invalidar sesiones activas y notificar.

**La Figura 4 muestra el caso de uso del RF-1.4: Recuperar Contraseña.**

Elemento	Descripción
<b>Actor</b>	Usuario registrado (o quien solicite recuperación).
<b>Precondición</b>	El usuario conoce el correo asociado (si existe).
<b>Secuencia Normal</b>	<ol style="list-style-type: none"> <li>1. Accede al flujo de recuperación.</li> <li>2. Ingresa correo.</li> <li>3. El sistema genera token/código y lo envía; registra evento.</li> <li>4. El usuario introduce el código o usa el enlace.</li> <li>5. El sistema valida token y permite cambio de contraseña.</li> <li>6. Se invalida el token y se notifica el cambio por correo.</li> </ol>
<b>Secuencia Alterna</b>	<ol style="list-style-type: none"> <li>A. Token expirado: pedir reenviar y bloquear si hay excesivos intentos.</li> <li>B. Código incorrecto: permitir reintentos limitados y luego bloquear.</li> </ol>
<b>Postcondición</b>	Contraseña actualizada; sesiones existentes revocadas.
<b>Excepciones</b>	Servicio de correo no disponible: encolar notificación y mantener token con estado ‘pending <sub>n</sub> otification’.
<b>Comentarios</b>	Registrar IP de origen y considerar desafío adicional si la solicitud se origina en país/dispositivo inusual.

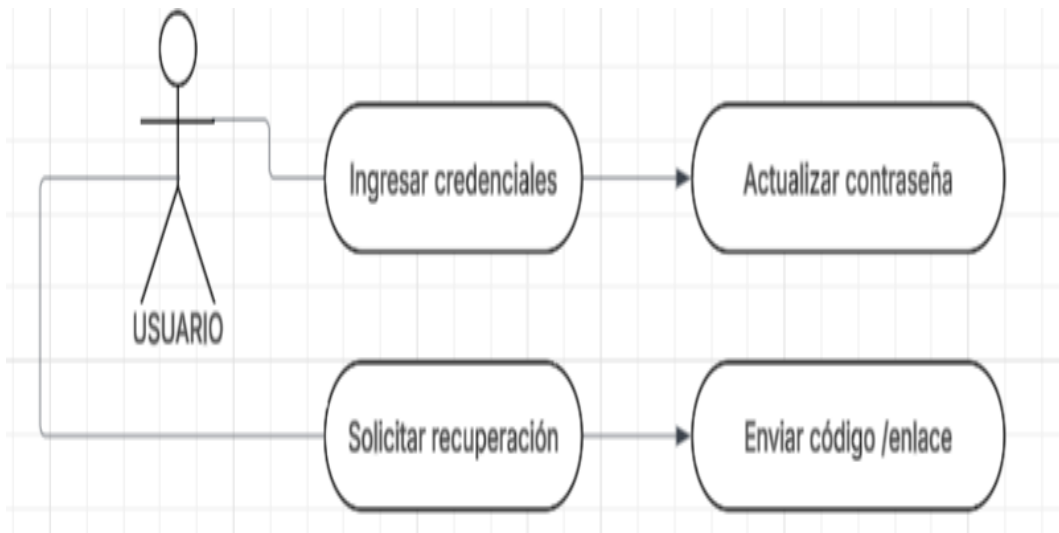


Figura 4: Diagrama del Caso de Uso – RF-1.4 Recuperar Contraseña. La figura muestra generación, envío y consumo de token/código.

## RF-1.5: Cerrar Sesión

### Descripción (extensa)

Funcionalidad para terminar la sesión del usuario y asegurar que no queden tokens válidos ni sesiones activas que puedan usarse indebidamente.

#### Requisitos técnicos:

- Invalide el refresh token en la base de datos y elimine cookies de sesión en el cliente.
- Registrar evento de logout con `user_id`, `timestamp`, `ip`.
- Si el backend usa JWT sin estado, mantener una blacklist revocable para tokens cuyo uso se desea impedir hasta expiración.

#### Casos especiales:

- Logout forzado por admin: administrador puede invalidar sesiones en masa.
- Cierre por expiración: el sistema detecta inactividad y finaliza sesión; informar al usuario sobre reinicio en el próximo acceso.

La Figura 5 muestra el caso de uso del RF-1.5: Cerrar Sesión.

Elemento	Descripción
<b>Actor</b>	Cliente, Asistente, Gerente (usuarios autenticados).
<b>Precondición</b>	Usuario tiene sesión activa o token válido.

<b>Secuencia Normal</b>	<ol style="list-style-type: none"> <li>1. El usuario selecciona “Cerrar Sesión”.</li> <li>2. El cliente solicita al servidor invalidar refresh token y limpiar cookies.</li> <li>3. El servidor marca token como revocado y registra evento de logout.</li> <li>4. El cliente redirige al inicio de sesión.</li> </ol>
<b>Secuencia Alterna</b>	Sesión ya expirada: redirigir al login sin error.
<b>Postcondición</b>	Sesión del usuario terminada y tokens invalidos.
<b>Excepciones</b>	Si el evento falla por problemas de red, mostrar mensaje y reintentar automáticamente; asegurar que no queden tokens activos en backend.
<b>Comentarios</b>	Garantizar que cierre de sesión invalide sesiones en todos los dispositivos (opción “Cerrar todas las sesiones”).

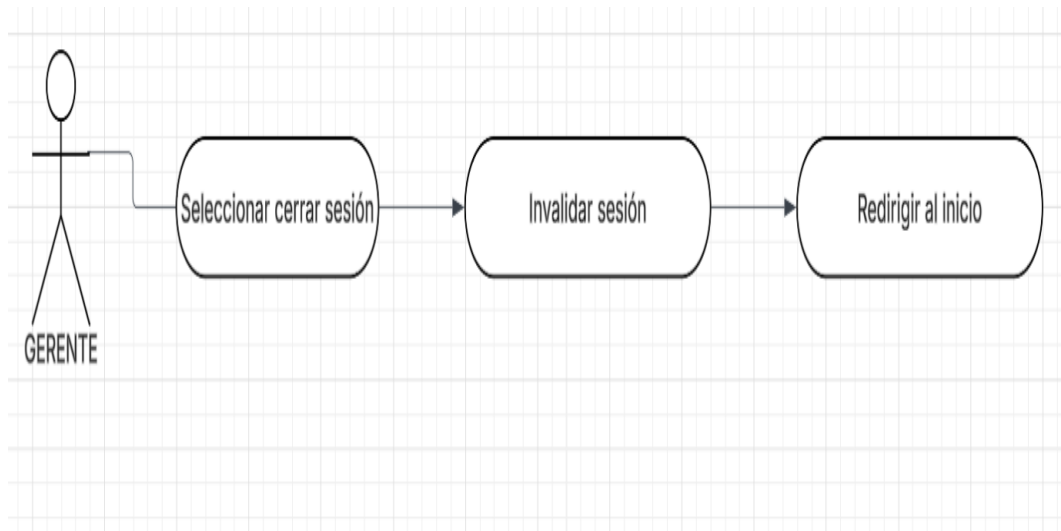


Figura 5: Diagrama del Caso de Uso – RF-1.5 Cerrar Sesión. La figura ilustra la invalidación de tokens y registro del evento.

## RF-1.6: Acceder al Dashboard

### Descripción (extensa)

Al autenticarse correctamente, cada actor será redirigido a un **Dashboard** personalizado que presenta información relevante para su rol, accesos rápidos a funcionalidades principales y widgets de estado. El Dashboard debe cargarse de forma segura, minimizando la superficie de ataque y evitando exponer datos no autorizados.

**Consideraciones de seguridad y rendimiento:**

- Las API que alimentan widgets deben validar token y permisos por endpoint.
- Cargar inicialmente sólo la información esencial y lazy-load para datos pesados.
- Implementar paginación y límites para llamadas a listados.
- Cifrar datos sensibles en los payloads cuando aplique y evitar renderizar PII innecesaria.

#### Widgets y ejemplos por rol:

- **Cliente:** estado de cuenta, balance, último pago, notificaciones.
- **Asistente:** tareas asignadas, citas pendientes, historial de consultas.
- **Gerente:** métricas clave, alertas de seguridad, accesos recientes, panel de control para gestión de usuarios.

**Accesos rápidos:** botones con acciones directas (ej. “Realizar Pago”, “Generar Reporte”) que verifican permisos y muestran confirmaciones antes de ejecutar operaciones críticas.

La Figura 6 muestra el caso de uso del RF-1.6: Acceder al Dashboard.

Elemento	Descripción
<b>Actor</b>	Cliente, Asistente, Gerente.
<b>Precondición</b>	Usuario autenticado y con token/ sesión válida.
<b>Secuencia Normal</b>	<ol style="list-style-type: none"> <li>1. Tras autenticación, el sistema identifica el rol del usuario.</li> <li>2. Redirige a la URL de su Dashboard correspondiente (p. ej. /dashboard/cliente).</li> <li>3. El Dashboard carga widgets resumen; cada widget solicita datos a API con autorización.</li> <li>4. El usuario interactúa con accesos rápidos y widgets; las acciones invocan casos de uso particulares.</li> </ol>
<b>Secuencia Alterna</b>	Si el token ha expirado al cargar data: forzar reautenticación y mostrar mensaje.
<b>Postcondición</b>	Usuario visualiza su página principal y puede acceder a funcionalidades clave.
<b>Excepciones</b>	Errores en APIs de widget: mostrar placeholders y permitir reintento manual.
<b>Comentarios</b>	Implementar monitoreo de latencia y errores en dashboards; usar CDN y cache donde aplique.

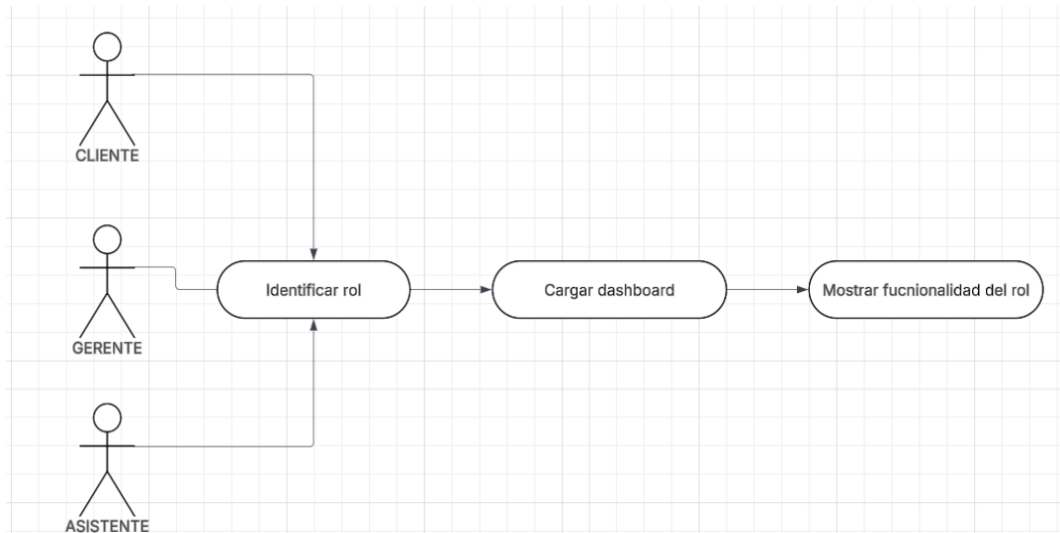


Figura 6: Diagrama del Caso de Uso – RF-1.6 Acceder al Dashboard. La figura muestra la identificación del rol y carga de widgets autorizados.