**Logotipo, nombre de la empresa

Descripción generada automáticamente**

Entrega Final

**INGENIERÍA DE SOFTWARE**

**Alumnos**: **Flores Z. Facundo**

**Localización**: Castelar

**Comisión**: 3-A

**Turno**: Noche

**Año** **de** **cursada**: 2024

**Docente**: **Julian Martin Rodriguez Escobedo**

Contents

[1. Historial de revisión 4](#_Toc173865722)

[2. Resumen 4](#_Toc173865723)

[3. Motivación 5](#_Toc173865724)

[4. Descripción funcional y alcance 5](#_Toc173865725)

[5. Información 6](#_Toc173865726)

[6. Especificaciones técnicas 7](#_Toc173865727)

[6.1.1 T01 – Arquitectura Base 7](#_Toc173865728)

[6.2 Diagrama 9](#_Toc173865729)

[6.2.1 Diagrama de arquitectura 9](#_Toc173865730)

[6.2.2 Diagrama de componentes 10](#_Toc173865731)

[6.2.3 Diagrama de secuencia – Persistencia de escritura 11](#_Toc173865732)

[6.2.4 Diagrama de secuencia – Persistencia de lectura 12](#_Toc173865733)

[6.2.5 Diagrama de entidad relación 13](#_Toc173865734)

[6.2.6 Diagrama de Clases 13](#_Toc173865735)

[7 T02 Gestión de Login/Logout y Gestión de Usuarios 14](#_Toc173865736)

[7.1 Objetivo 14](#_Toc173865737)

[7.2 Descripción del Funcionamiento 14](#_Toc173865738)

[7.3 Diagrama de secuencia de LogIn 15](#_Toc173865739)

[7.4 Diagrama de secuencia de LogOut 15](#_Toc173865740)

[8. T03. Gestión de Encriptado 16](#_Toc173865741)

[8.1 Objetivo 16](#_Toc173865742)

[8.2 Descripción del Funcionamiento 16](#_Toc173865743)

[9. T05. Gestión de Múltiples Idiomas 16](#_Toc173865744)

[9.1 Objetivo 16](#_Toc173865745)

[9.2 Descripción del Funcionamiento 16](#_Toc173865746)

[10. T06a Gestión de Bitácora 17](#_Toc173865747)

[10.1 Objetivo 17](#_Toc173865748)

[10.2 Descripción del Funcionamiento 17](#_Toc173865749)

[10.2.1 Registro de Actividades 17](#_Toc173865750)

[10.2.2 Filtrado de Eventos 17](#_Toc173865751)

[10.3 Diagrama secuencia Bitácora 18](#_Toc173865752)

[10.4 Diagrama de secuencia Persistencia Escritura 18](#_Toc173865753)

[11. T06b. Control de Cambios 19](#_Toc173865754)

[11.1 Objetivo 19](#_Toc173865755)

[11.2 Descripción del Funcionamiento 19](#_Toc173865756)

[11.3 Diagrama de secuencia de gestión de permisos (Composite) 20](#_Toc173865757)

[12 T07. Gestión de Dígito Verificador 20](#_Toc173865758)

[12.1 Objetivo 20](#_Toc173865759)

[12.2 Dígito Verificador Horizontal (DVH) 20](#_Toc173865760)

[12.3 Dígito Verificador Vertical (DVV) 21](#_Toc173865761)

[12.4 Diagrama de secuencia digito verificador 22](#_Toc173865762)

# Historial de revisión

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Alumno** | **Descripción** |
| 25/04/2024 | 1.0 | Flores Z. Facundo | Creación del documento |
| 28/04/2024 | 1.1 | Pablo N. Montero | Realización de diagramas |
| 01/5/2024 | 1.2 | Pablo N. Montero | Realización de diagrama de secuencia Bitácora y secuencia Login |
| 02/5/2024 | 1.3 | Flores Z. Facundo | Realización de diagrama de Componentes |
| 02/5/2024 | 1.4 | Flores Z. Facundo | Realización de diagrama de secuencia Logout, persistencia Lectura |
| 01/5/2024 | 1.5 | Pablo N. Montero | Realización de diagrama de secuencia de persistencia escritura y de Clases |
| 11/07/2024 | 1.6 | Flores Z. Facundo | actualización Multi Idioma |
| 05/08/2024 | 1.7 | Flores Z. Facundo | Actualización del documento/Portada |

# Resumen

En esta primera fase del proyecto práctico se abordan varios aspectos cruciales para la construcción de un sistema sólido y seguro. Se utilizará una arquitectura de base compuesta por seis capas, lo que proporcionará una estructura organizada y modular que facilitará el desarrollo de la aplicación.

La gestión de inicio y cierre de sesión (Login y Logout) de usuarios es esencial para la seguridad de la información y será implementada en esta etapa inicial. También se incluirá la gestión de registros para documentar las acciones realizadas por los usuarios y monitorear la aplicación en tiempo real (Bitácora).

Además, se emplea el cifrado hash para proteger las contraseñas de los usuarios, garantizando así la seguridad de sus datos personales.

Esta primera fase del proyecto sentará los cimientos para el desarrollo de una aplicación segura, modular y robusta, aprovechando todos los conocimientos adquiridos durante la carrera para garantizar su éxito.

# Motivación

La principal motivación detrás del desarrollo de este sistema radica en la importancia de mejorar tanto la eficiencia como la seguridad en la gestión de información en diversas áreas. La automatización de procesos y la reducción de errores humanos pueden generar notables beneficios en cuanto a productividad y rentabilidad.

La protección de la información personal y financiera de los usuarios es cada vez más crucial. Por ende, la gestión de la seguridad de los datos debe ser una prioridad indiscutible en cualquier aplicación o sistema.

Por otro lado, la elección de la arquitectura de 6 capas se fundamenta en su capacidad para proporcionar una estructura organizada y escalable, lo que facilita el desarrollo y mantenimiento de la aplicación con mayor eficacia.

# Descripción funcional y alcance

El sistema desarrollado tiene como propósito primordial ofrecer una solución segura y eficaz para la gestión de información en distintos ámbitos.

Los usuarios tienen la capacidad de acceder al sistema mediante sus credenciales, las cuales son validadas a través de una base de datos segura. En caso de que el usuario no esté registrado, se le permite hacerlo, asegurando que su contraseña cumpla con las expresiones regulares establecidas para garantizar la seguridad de sus datos.

Una vez dentro del sistema, los usuarios pueden explorar el menú de la aplicación y llevar a cabo diversas acciones. Todas las interacciones realizadas por los usuarios son registradas en la base de datos para mantener un historial y monitorear el sistema en tiempo real.

El sistema implementa una función de cierre de sesión que permite a los usuarios salir en cualquier momento para asegurar la protección de los datos y protegerla información personal.

El sistema desarrollado proporciona una solución segura y eficiente para la gestión de información en diversas áreas. Los usuarios acceden mediante credenciales seguras, y el sistema registra todas las operaciones realizadas. Además, se garantiza la seguridad de los datos con la función de cierre de sesión.

1. Verificación de credenciales de acceso: Este proceso de negocio consiste en chequear las credenciales de acceso del usuario (como el nombre de usuario y la contraseña) contra la base de datos del sistema para confirmar si el usuario tiene permiso para acceder al sistema. Si las credenciales no son correctas, se rechaza el acceso.
2. Manejo de sesiones de usuario: Después de que el usuario inicia sesión, el sistema crea una sesión que le permite interactuar con diversas funciones. El manejo de sesiones incluye la identificación y el seguimiento de la sesión activa, además de su eliminación cuando el usuario se desconecta.
3. Control de actividades del usuario: El sistema rastrea el inicio y cierre de sesión del usuario, registrando las fechas de ingreso y egreso correspondientes. Esta información ayuda a monitorear el sistema y a mantener un registro de las acciones realizadas.
4. Proceso de cierre de sesión: Este proceso se refiere a la terminación de la sesión activa del usuario cuando se desconecta del sistema. Al cerrar la sesión, se asegura la protección de los datos del usuario y se evita el acceso no autorizado.
5. Registro de usuarios: Este proceso implica crear un perfil para el usuario en la base de datos del sistema, que incluye las credenciales de acceso. También se verifica que la contraseña cumpla con las reglas de seguridad establecidas. Este proceso puede realizarse mediante un formulario de registro previo al ingreso al sistema.

# Información

Para la entrega el documento contendrá detalles sobre la arquitectura utilizada en el sistema, incluyendo las razones para elegir esta arquitectura específica y cómo se adapta a los objetivos y alcance del sistema.

Además, se incluirán varios diagramas de secuencia para ilustrar los procesos de inicio y cierre de sesión de los usuarios. Estos diagramas explicarán cómo interactúa el usuario con el sistema y cómo se lleva a cabo la verificación de credenciales y el cifrado de la contraseña.

El documento también contendrá un diagrama de clases que ilustrará la estructura y relaciones entre los diversos objetos y componentes del sistema. Este diagrama ayudará a entender claramente cómo se modelaron los distintos elementos del sistema y cómo interactúan entre sí.

Un diagrama entidad-relación (DER) también será parte del documento, mostrando las distintas entidades y relaciones en la base de datos del sistema. Este diagrama facilitará la comprensión del diseño de la base de datos y de cómo se vinculan las diferentes tablas.

Finalmente, se incluirá un diagrama de componentes que muestre los diversos componentes del sistema y cómo interactúan entre ellos. Este diagrama permitirá entender cómo se dividió el sistema en componentes y cómo se comunican para brindar la funcionalidad total del sistema.

# Especificaciones técnicas

## 6.1.1 T01 – Arquitectura Base

El proyecto desarrollado adopta una arquitectura de 6 capas, utilizando Visual Studio como entorno de desarrollo. Esta elección se basa en requerimientos y necesidades específicas del proyecto. Las capas en esta arquitectura incluyen: Aplicación (Entidades de Negocio), Dominio (Lógica de Negocio), DAL (Acceso a Datos), UI (Interfaz de Usuario), Abstracción y Servicios.

Se describen las funciones de cada capa y el motivo de su inclusión en el proyecto:

Aplicación (Entidades de Negocio): Define las estructuras de datos que se utilizan en la aplicación, representando objetos de negocio. Esta capa abstrae el detalle de los datos para que la lógica de negocio pueda centrarse en las necesidades empresariales sin preocuparse por la implementación subyacente.

Dominio (Lógica de Negocio): Encargada de la lógica empresarial, esta capa define las reglas y cómo se aplican a las entidades definidas en la capa de Aplicación . Aquí se implementan los procesos de negocio y las reglas para la manipulación de datos.

DAL (Acceso a Datos): Responsable del acceso y almacenamiento de datos. Esta capa define la forma de interactuar con la base de datos y usa las entidades de la capa de Aplicación para guardar y recuperar datos.

UI (Interfaz de Usuario): La capa de interfaz de usuario gestiona la presentación de la aplicación y las interacciones con el usuario. Se basa en la capa de Dominio para realizar operaciones, y a través de ella, el usuario interactúa con el sistema.

Abstracción: Proporciona una capa intermedia entre Dominio y DAL, permitiendo la traducción entre las entidades definidas en Aplicación y los objetos usados por DAL. Esto ayuda a mantener una separación clara de responsabilidades y facilita el mantenimiento del código.

Servicios: Esta capa se encarga de proporcionar servicios a otras aplicaciones o sistemas, definiendo cómo se accede a ellos y qué tipo de servicios se ofrecen.

Las razones para optar por una arquitectura de 6 capas incluyen:

Separación de responsabilidades: Cada capa tiene un propósito claro, lo que hace que el código sea más modular y fácil de mantener. Al tener responsabilidades definidas, el código es más fácil de entender y gestionar.

Reutilización del código: Con capas específicas, es posible reutilizar segmentos de código, ahorrando tiempo y esfuerzo en el desarrollo. Los módulos se pueden utilizar en diferentes partes de la aplicación.

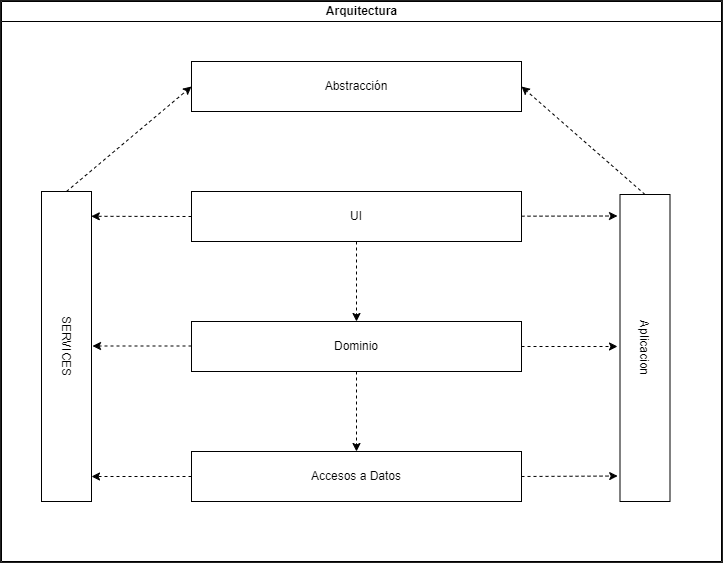
Flexibilidad y escalabilidad: Cada capa se puede modificar o mejorar sin afectar las demás, permitiendo agregar nuevas funcionalidades sin reescribir todo el código.

Facilidad para pruebas: La arquitectura de 6 capas facilita la realización de pruebas unitarias y de integración, lo que ayuda a identificar errores tempranamente y mejorar la calidad del software.

Facilidad para el mantenimiento: Como cada capa tiene una función específica, es más sencillo localizar y corregir errores en el código, simplificando el mantenimiento.

# 6.2 Diagrama

## 6.2.1 Diagrama de arquitectura



## 6.2.2 Diagrama de componentes

Diagrama

Descripción generada automáticamente

## 6.2.3 Diagrama de secuencia – Persistencia de escritura

Diagrama

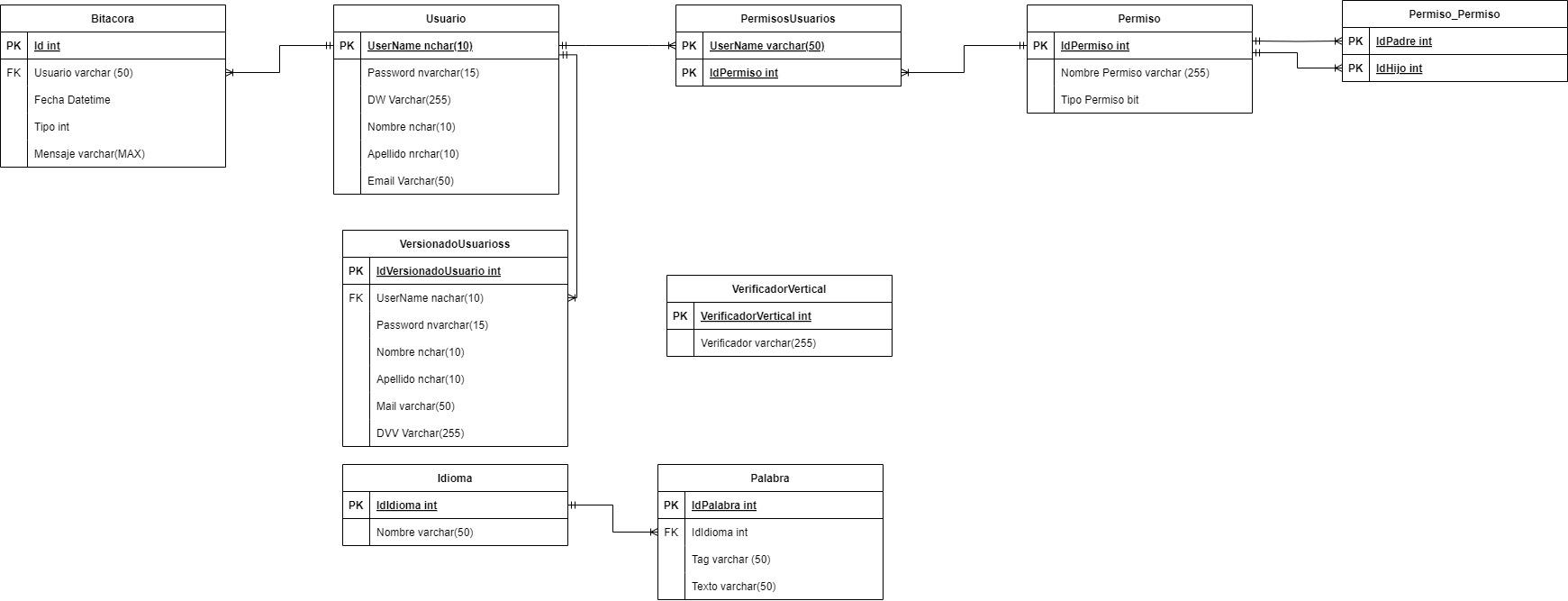
Descripción generada automáticamente

## 6.2.4 Diagrama de secuencia – Persistencia de lectura

Diagrama

Descripción generada automáticamente

## 6.2.5 Diagrama de entidad relación



## 6.2.6 Diagrama de Clases

Diagrama, Esquemático

Descripción generada automáticamente

# 7 T02 Gestión de Login/Logout y Gestión de Usuarios

## 7.1 Objetivo

El propósito de la funcionalidad de gestión de inicio/cierre de sesión y administración de usuarios es verificar la identidad de los usuarios mediante el ingreso de su nombre de usuario y contraseña. La función principal es proporcionar al usuario un acceso seguro y asignarle los permisos necesarios para gestionar el sistema dentro de sus límites.

## 7.2 Descripción del Funcionamiento

**Proceso de Inicio de Sesión**

El proceso de inicio de sesión comienza cuando un usuario ingresa sus datos de usuario y contraseña en el formulario de acceso. Primero, se verifica que los campos de usuario y contraseña estén completos. Si los campos no están vacíos, se procede a encriptar el nombre de usuario y realizar un hash sobre la contraseña. Con estos datos, se verifica la existencia del usuario en la base de datos SQL.

Si el usuario existe y las credenciales son correctas, se le otorgará acceso y se cargarán en el sistema los permisos correspondientes, manteniendo la integridad de sus datos. Además, se registrará el evento en la bitácora. Si las credenciales son incorrectas, se informará al usuario mediante un mensaje y se registrará el evento en la bitácora.

**Proceso de Cierre de Sesión**

Una vez que el usuario finalice la sesión, será redirigido a la interfaz de inicio de sesión.

## 7.3 Diagrama de secuencia de LogIn

Diagrama

Descripción generada automáticamente

## 7.4 Diagrama de secuencia de LogOut

Diagrama

Descripción generada automáticamente

# 8. T03. Gestión de Encriptado

## 8.1 Objetivo

El propósito de la gestión de encriptado es implementar algoritmos de cifrado para proteger los datos sensibles del sistema. Se utilizan tanto algoritmos de cifrado simétrico como asimétrico, con el objetivo final de asegurar la protección de los datos sensibles del sistema.

## 8.2 Descripción del Funcionamiento

**Algoritmos de Cifrado**

Se ha implementado un algoritmo de cifrado simétrico utilizando MD5 para el nombre de usuario, y un algoritmo de cifrado hash para almacenar las contraseñas de los usuarios.

# 9. T05. Gestión de Múltiples Idiomas

## 9.1 Objetivo

El objetivo de la gestión de múltiples idiomas, utilizando el patrón Observer, es permitir que un sistema pueda adaptarse y ofrecer soporte para diferentes idiomas de manera dinámica. Se busca separar la lógica de gestión de idiomas de la interfaz de usuario y asegurar que los cambios de idioma se propagan automáticamente a todas las partes del sistema que necesiten traducción.

## 9.2 Descripción del Funcionamiento

**Modelo de Gestión de Idiomas**

Se implementa un modelo de gestión de idiomas que es independiente de la interfaz de usuario. Esto significa que la lógica de traducción y gestión de idiomas reside en un componente separado, lo cual facilita su reutilización en diferentes partes del sistema.

**Patrón Observer**

Se utiliza el patrón Observer para establecer una relación de suscriptor-observable entre el componente de gestión de idiomas y la traducción. El componente de gestión de idiomas actúa como el observable, mientras que los formularios del sistema que requieren traducción actúan como suscriptores.

**Registro de Suscriptores**

El componente de gestión de idiomas mantiene una lista de suscriptores, que son las partes del sistema interesadas en recibir notificaciones sobre los cambios de idioma. Cada suscriptor se registra en el componente de gestión de idiomas.

**Cambio de Idioma**

Cuando se produce un cambio de idioma, el componente de gestión de idiomas actualiza internamente el idioma seleccionado y notifica a todos los suscriptores registrados.

# 10. T06a Gestión de Bitácora

## 10.1 Objetivo

La gestión de la bitácora tiene como meta registrar todas las operaciones realizadas por los usuarios dentro del sistema, permitiendo así un control detallado de las actividades. La bitácora almacena información como el usuario, la fecha en que se registró el evento, el tipo de registro y una descripción clara y referencial de lo ocurrido.

## 10.2 Descripción del Funcionamiento

### 10.2.1 Registro de Actividades

La funcionalidad de gestión de la bitácora se encarga de documentar las actividades realizadas por los usuarios durante el uso del sistema. Por cada acción relevante se genera un evento en la bitácora. Este evento incluye información esencial, como la fecha, el usuario, el tipo de registro y una breve descripción de la actividad.

### 10.2.2 Filtrado de Eventos

La principal idea es que los eventos registrados en la bitácora se puedan filtrar según criterios específicos, como un rango de fechas y los usuarios que realizaron una actividad. Esto facilita la búsqueda de información específica por cada usuario y la actividad que realizó.

## Diagrama secuencia Bitácora

Diagrama

Descripción generada automáticamente

## 10.4 Diagrama de secuencia Persistencia Escritura

Diagrama

Descripción generada automáticamente

# 11. T06b. Control de Cambios

## 11.1 Objetivo

El objetivo del control de cambios es ofrecer una funcionalidad de auditoría que permita realizar un seguimiento detallado de todas las modificaciones efectuadas durante el ciclo de vida de una entidad en el sistema. Esta funcionalidad busca gestionar cada alteración, proporcionando un historial exhaustivo y la capacidad de restaurar el estado previo de un objeto específico.

## 11.2 Descripción del Funcionamiento

**Registro de Cambios**

Cada vez que se realiza una modificación en el objeto de una entidad, se registra una entrada en el historial de cambios. Esta entrada incluye información sobre las distintas modificaciones que ha experimentado el objeto a lo largo del tiempo.

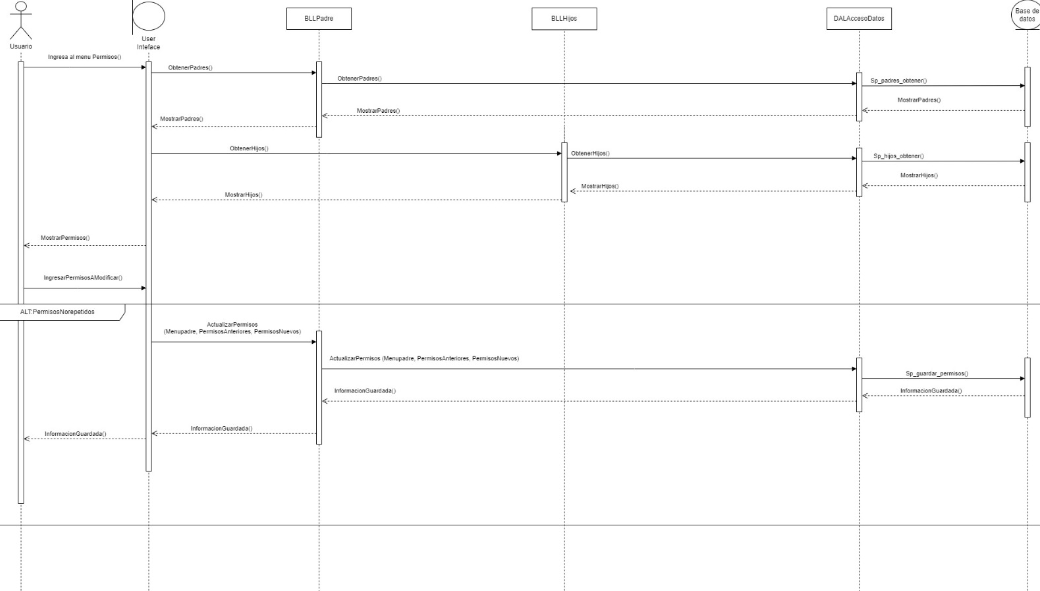
**Consulta del Historial**

El historial de cambios está accesible para visualizar la lista de modificaciones realizadas en orden cronológico. Siempre aparecerá la última versión disponible para restaurar.

**Restauración del Estado Anterior**

Se implementa una funcionalidad que permite deshacer los cambios realizados en el objeto y restaurar su estado anterior. Se puede seleccionar una modificación específica del historial y restaurar el estado previo del objeto.

## 11.3 Diagrama de secuencia de gestión de permisos (Composite)



# 12 T07. Gestión de Dígito Verificador

## 12.1 Objetivo

El objetivo de la gestión de dígitos verificadores es asegurar la integridad de los datos almacenados en la base de datos. Esto permite detectar modificaciones no autorizadas, ya sea agregando, eliminando o intercambiando datos. Se utilizan dígitos verificadores horizontales y verticales para este propósito.

## 12.2 Dígito Verificador Horizontal (DVH)

El DVH verifica la integridad de los datos almacenados en un atributo de una entidad. Se almacena en un atributo específico de la entidad. El cálculo del DVH incluye tanto el contenido del atributo como su posición y la posición del atributo dentro de la entidad.

Para generar el DVH, se concatenan atributos como nombre, apellido, email, dirección y contraseña del usuario. Luego, se aplica un algoritmo de hash al resultado obtenido. El valor resultante se guarda en el atributo DVH de la tabla correspondiente.

Para verificar la integridad de los datos, se recalcula el DVH utilizando los mismos atributos y algoritmo de hash, y se compara el valor calculado con el valor almacenado en el atributo DVH. Si coinciden, los datos son considerados íntegros; de lo contrario, se detecta una posible alteración en los datos.

## 12.3 Dígito Verificador Vertical (DVV)

El DVV verifica la integridad de los DVH en una entidad y se almacena en una entidad adicional creada para este propósito.

Para generar el DVV, se concatena los valores de todos los DVH en la entidad y se aplica un algoritmo de hash al resultado obtenido. El valor resultante se guarda como DVV en la entidad correspondiente.

Para verificar la integridad de los DVH, se recalcula el DVV utilizando los DVH actuales y se compara el valor calculado con el valor almacenado en la entidad de DVV. Si coinciden, los DVH se consideran íntegros; de lo contrario, se detecta una posible alteración en los datos.

**Proceso de Verificación de Integridad al Iniciar la Aplicación**

Antes de permitir el acceso al sistema, se realiza un proceso de verificación de integridad de la base de datos que incluye la verificación tanto de los DVH como del DVV. Si se detecta algún dato corrupto en la base de datos, se mostrará en pantalla un mensaje y no podrá continuar hasta que no se solucione la integridad de los datos.

## 12.4 Diagrama de secuencia digito verificador

Imagen que contiene Tabla

Descripción generada automáticamente