

***Facultad***

***de***

***Ciencias***

**DESARROLLO DE PLATAFORMA WEB DE GESTIÓN DE ADAPTACIONES PARA LA EBAU**

(Development of a web platform to manage adaptations for the EBAU (University Entrance Examination))

Trabajo de Fin de Grado

para acceder al

**GRADO EN INGENIERÍA INFORMÁTICA**

Autor: Yago Nava Bolado

Director: Diego García Saiz

Co-Director: Javier Muñoz de la Iglesia

Junio – 2024

ÍNDICE

[INTRODUCCIÓN Y OBJETIVOS 3](#_Toc168298436)

[Introducción en castellano 3](#_Toc168298437)

[Objetivos 5](#_Toc168298438)

[METODOLOGÍA Y CALENDARIO 7](#_Toc168298439)

[SOFTWARE UTILIZADO 10](#_Toc168298440)

[REQUISITOS FUNCIONALES Y NO FUNCIONALES 11](#_Toc168298441)

[Requisitos Funcionales: 11](#_Toc168298442)

[Requisitos no funcionales: 11](#_Toc168298443)

[CASOS DE USO 12](#_Toc168298444)

[ARQUITECTURA 16](#_Toc168298445)

[Descripción de Componentes y Conexiones: 16](#_Toc168298446)

[Base de datos: 16](#_Toc168298447)

[Back-End: 20](#_Toc168298448)

[Patrón de Diseño: 21](#_Toc168298449)

[Escalabilidad y Seguridad: 22](#_Toc168298450)

[Herramientas y Recursos: 23](#_Toc168298451)

[DESARROLLO, DISEÑO E IMPLEMENTACIÓN 23](#_Toc168298452)

[Diseño de Base de Datos 23](#_Toc168298453)

[Modelado de ER (Modelo Relacional) 23](#_Toc168298454)

[Paso a tablas 23](#_Toc168298455)

[Diseño e implementación de procedimientos almacenados (Lógica de Negocio) 23](#_Toc168298456)

[Diseño de Modelo de Dominio 23](#_Toc168298457)

[Desarrollo de API (Interfaz de Programación de Aplicaciones) 23](#_Toc168298458)

[Implementación de Modelo de Dominio 23](#_Toc168298459)

[Conexión con Base de datos 23](#_Toc168298460)

[Conexión Back-End – Front-End 23](#_Toc168298461)

[Implementación de Interfaz Gráfica 23](#_Toc168298462)

[Configuración de la API 23](#_Toc168298463)

[PRUEBAS: UNITARIAS, INTEGRACIÓN Y ACEPTACIÓN 24](#_Toc168298464)

[EJEMPLO DE USO 24](#_Toc168298465)

[CONCLUSIONES 24](#_Toc168298466)

[TRABAJO FUTURO 24](#_Toc168298467)

[TRABAJO PREVIO 24](#_Toc168298468)

[FASES DE DESARROLLO 24](#_Toc168298469)

# INTRODUCCIÓN Y OBJETIVOS

## Introducción en castellano

Gestionar las adaptaciones extraordinarias para estudiantes de 2º de Bachillerato durante la prueba de acceso a la universidad representa un desafío complejo para el Servicio de Orientación Universitaria de Cantabria (SOUCAN).Actualmente, este proceso se realiza a través de comunicaciones por correo electrónico con los centros educativos de Cantabria, que detallan la necesidad de una serie de adaptaciones para su alumnado dentro de un listado existente proporcionado por el SOUCAN, y adjuntan todos los documentos solicitados. Estos documentos se registran en un extenso archivo de Excel, cuya lectura y edición resulta muy complicada debido a su dimensión.

Para abordar eficientemente esta necesidad, se propuso el desarrollo integral de una aplicación web, es decir, que su elaboración involucre todos los elementos que se incluyen en la construcción de una aplicación software. Esta herramienta facilitará y automatizará la gestión de tareas tanto al departamento de la Universidad de Cantabria como a los centros educativos de la región en la prestación de este servicio.

El proceso de desarrollo se divide en varias etapas clave, comenzando por el diseño del dominio y los objetos que requiere la aplicación. Esto es crucial ya que implica la identificación y definición de los conceptos esenciales del sistema, así como las relaciones entre ellos. Para realizar este proceso, se utiliza la herramienta “Umlet”, la cual proporciona una manera gráfica de representar el diagrama UML a crear.

Posteriormente, se procede con el diseño de la base de datos. Esta fase implica definir la estructura de la base de datos, incluyendo las tablas, relaciones y restricciones necesarias para almacenar de manera eficiente y segura toda la información relevante para la gestión de adaptaciones para los estudiantes, vinculada con el dominio diseñado previamente.

El desarrollo del Back-End será fundamental para el correcto funcionamiento del sistema. La lógica de negocio se desarrolla a través de una serie de procedimientos almacenados en la base de datos, lo que permitirá un uso seguro y eficiente para el almacenamiento y gestión de datos, permitiendo realizar operaciones de recuperación, inserción, modificación y eliminación de datos. Posteriormente, se establece la conexión de la API con la base de datos a través del mapeado de estos procedimientos, implementando los mecanismos necesarios para que la aplicación pueda acceder y manipular la información y relacionarse con los objetos manejados por la interfaz.

En cuanto al desarrollo del Front-End, se lleva a cabo la creación de la interfaz gráfica de usuario, que será la parte visible y accesible de la aplicación para los usuarios finales. Este proceso se realiza mediante el uso de vistas y controladores para obtener la apariencia y funcionalidades deseadas. La interfaz se diseña de manera intuitiva, con el objetivo de facilitar la interacción de los usuarios con el sistema y proporcionar una experiencia satisfactoria. Es importante destacar que esta funcionalidad estará disponible exclusivamente para los centros educativos, permitiendo el uso remoto del servicio. Por otro lado, para el departamento del SOUCAN, se utiliza la herramienta “Microsoft Access.". Esto se debe a que permite realizar cambios dentro de la base de datos de manera local y proporciona una funcionalidad de gestión completa, siendo administrada desde las instalaciones de la Universidad de Cantabria.

Para mencionar las tecnologías utilizadas, se emplea "SQL Server" para el diseño del almacenamiento de datos, "Visual Studio" junto con el lenguaje C# y el framework ".NET Framework" para el desarrollo de la API, y "Microsoft Access" para implementar las funcionalidades que serán utilizadas desde la Universidad de Cantabria. Todo el proyecto será recogido como Software libre en GitHub.

Introducción en inglés

Managing extraordinary adaptations for 12th-grade students during the university entrance exam represents a complex challenge for the University Orientation Service of Cantabria (SOUCAN). Currently, this process is carried out through email communications with educational centers in Cantabria, which explain the need for a series of adaptations for their students within an existing list provided by SOUCAN and attach all the requested documents to register them in an extensive Excel document, which is very complex to read/edit due to its size.

To effectively address this need, the development of a web application in its entirety was proposed, meaning that its development involves all the elements included in the construction of a software application. This tool will facilitate and automate task management for both the University of Cantabria department and the provision of services to educational centers in the region.

The development process is divided into several key stages, starting with the design of the domain objects required for the application, as this is crucial since it involves the identification and definition of the essential system concepts, as well as the relationships between them. To carry out this process, the "Umlet" tool was used, which provides a graphical way to represent the UML diagram to be created.

Subsequently, the database design is carried out. This phase involves defining the structure of the database, including the tables, relationships, and constraints necessary to efficiently and securely store all the relevant information for managing adaptations for students, which is related to the previously designed domain.

Back-End development will be essential for the proper functioning of the system. Business logic will be developed in a series of stored procedures in the database, which will allow safe and efficient use of it for data storage and management, allowing for data retrieval, insertion, modification, and deletion operations. Subsequently, the API connection with the database will be established through the mapping of these procedures, implementing the necessary mechanisms for the application to access and manipulate the information and interact with the objects it will work with the interface.

Regarding Front-End development, the creation of the user interface will be carried out, which will be the visible and accessible part of the application for end users. This process is carried out using views and controllers to obtain the desired appearance and functionality. The interface is designed intuitively, with the aim of facilitating user interaction with the system and providing a satisfactory experience. It is important to note that this functionality will be available exclusively to educational centers, allowing remote use of the service. On the other hand, for the SOUCAN department, the Microsoft Access application will be used. This is because it allows changes to be made within the database locally and provides complete management functionality, being managed from the facilities of the University of Cantabria.

To mention the technologies used, "SQL Server" will be used for data storage design, "Visual Studio" along with the C# language and the ".NET Framework" for API development, and "Microsoft Access" to implement the functionalities that will be used from the University of Cantabria. The entire project will be collected as Open-Source Software on GitHub.

## Objetivos

Los objetivos que cumplir en este proyecto son muy concretos, ya que lo que trata es de facilitar y automatizar una tarea ya existente dentro del departamento del SOUCAN. Todo este procedimiento se puede dividir en varias etapas, las cuales serán realizadas de una manera más sencilla. A continuación, se realiza un contraste entre el proceso actual y el nuevo:

* Contacto SOUCAN/Centros:
  + En la actualidad: La comunicación y las solicitudes se gestionan exclusivamente a través del correo electrónico, lo que puede generar malentendidos y falta de claridad entre las partes involucradas.
  + Con la implementación de la aplicación: El uso del correo electrónico se limita únicamente al momento en que los centros solicitan un usuario para acceder al servicio. A partir de entonces, la aplicación proporciona una plataforma intuitiva y sencilla que facilita la comunicación y gestión de todas las solicitudes entre ambas partes.
* Disponibilidad de Asignaturas:
  + En la actualidad: Los centros se informan sobre las asignaturas disponibles para que sus alumnos puedan realizar la EBAU consultando la página web de la Universidad. Luego, notifican al SOUCAN sobre las asignaturas para las cuales prevén que los estudiantes se presenten, con el fin de determinar qué aulas pueden requerir adaptaciones.
  + Con la implementación de la aplicación: Se tiene la capacidad de gestionar las asignaturas que son visibles en el formulario de registro desde el apartado de Access. Esto permite mantener actualizado y estructurado el listado de asignaturas disponibles para cada convocatoria, dividido en las distintas fases con las que cuenta la EBAU. Como resultado, los centros pueden realizar el registro de asignaturas de manera más sencilla y automática, sin necesidad de consultar la página web de la Universidad.
* Disponibilidad de Adaptaciones para diagnósticos:
  + En la actualidad: La organización de las adaptaciones disponibles para estudiantes con diagnósticos específicos se gestiona mediante la selección de las adaptaciones pertinentes para cada caso individual en función de la solicitud que se realice. Esto provoca que el proceso resulte tedioso y complejo, por lo que debería estar automatizado para agilizar el trabajo.
  + Con la implementación de la aplicación: Se simplifica y acelera este proceso al estructurarlo de manera que, antes de abrir los plazos de registro, se establecen las adaptaciones disponibles para cada posible diagnóstico. Esto estandariza el proceso y permite que todos los estudiantes puedan realizar la solicitud de manera más intuitiva sin dar lugar a confusiones.
* Asignación de Sedes de EBAU para centros:
  + En la actualidad: La universidad comunica a los centros la ubicación donde se llevará a cabo la EBAU mediante correo electrónico. Esto significa que los centros deben documentar esta información para evitar extravíos y la necesidad de solicitar recordatorios de esta.
  + Con la implementación de la aplicación: Dentro del servicio, el SOUCAN tiene la capacidad de asignar o actualizar la sede de cada centro. Esto garantiza que los centros tengan acceso a esta información de manera rápida y sin riesgo de perderla. De esta manera, se elimina la necesidad de que los centros documenten activamente esta información, ya que pueden confiar en que estará disponible y actualizada en el servicio proporcionado.
* Organización y Disponibilidad de Documentación sobre Estudiantes:
  + En la actualidad: La recopilación de los documentos necesarios para que un estudiante pueda solicitar las adaptaciones necesarias durante el examen se vuelve compleja debido al volumen de archivos almacenados y su distribución en varias carpetas. Esta organización de archivos hace que sea tedioso encontrar documentos específicos de estudiantes en concreto en caso de necesitar revisarlos. Además, existe el riesgo de que los centros adjunten archivos incorrectos, lo que requiere que el SOUCAN se ponga en contacto, solicite el cambio de documento, elimine el incorrecto y finalmente almacene el correcto.
  + Con la implementación de la aplicación: Se ha incorporado una funcionalidad que, al rellenar el formulario, solicita el documento necesario, permitiendo que sea renombrado de manera estandarizada. Esto facilita la búsqueda del documento cuando sea necesario. Además, el SOUCAN tiene la capacidad de revisar la información de un estudiante registrado y visualizar sus documentos. En caso de ser necesario, puede validarlos o rechazarlos, lo que indica a los centros que deben realizar el cambio de archivo correspondiente a través del servicio.
* Documentación acerca de Reuniones sobre Estudiantes:
  + En la actualidad: En ocasiones, la persona encargada de esta gestión en la universidad debe reunirse tanto con los estudiantes como con los coordinadores de los centros. Es necesario documentar estas reuniones, ya que proporcionan información relevante sobre los estudiantes que dependen de este servicio universitario. Sin embargo, esta forma de organización puede dificultar la localización de los archivos concluyentes debido al gran número de estudiantes involucrados.
  + Con la implementación de la aplicación: A través del servicio, en la parte gestionada por el SOUCAN, es posible recopilar la información de las reuniones en texto plano. Esto facilita la ubicación de la documentación, ya que está directamente asociada al estudiante correspondiente dentro de la base de datos. Además, mejora la eficiencia del almacenamiento, ya que, en lugar de guardar documentos completos, la información se almacena en formato de texto.
* Persistencia de Datos:
  + En la actualidad: Toda la recopilación de las distintas convocatorias se almacena de forma independiente, lo que dificulta realizar un análisis estadístico unificado. Esta tarea se vuelve compleja y lenta, ya que requiere revisar múltiples documentos y apuntar los datos por separado para poder realizar la comparativa.
  + Con la implementación de la aplicación: Esta tarea se simplifica considerablemente al utilizar el servicio, que cuenta con una sección dedicada a realizar operaciones con toda la información recopilada en la base de datos. Además, el registro de cada estudiante está vinculado a la convocatoria activa en ese momento, lo que facilita la comparación de datos. Esta sección está diseñada específicamente para ser implementada, aunque por el momento no se conoce qué tipo de contraste de datos se desea realizar.

# METODOLOGÍA Y CALENDARIO

En este apartado se detalla el enfoque y las etapas necesarias para el desarrollo del servicio. Se describirán las fases del ciclo de desarrollo, desde la planificación y el diseño hasta la implementación y las pruebas, así como la gestión de tiempos y recursos. Además, se presentará un informe con los hitos logrados en cada plazo, lo que permitirá un seguimiento estructurado del progreso, asegurando la consecución de los objetivos propuestos de manera eficiente y organizada.

Se adjunta su debido Diagrama de Gantt, que permite representar de manera gráfica cómo ha sido subdividido en el tiempo el desarrollo de las tareas:

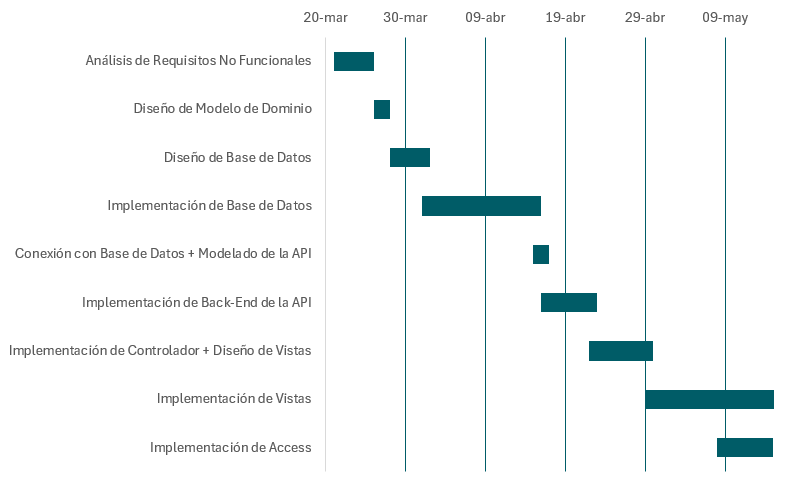


Tabla 1: Diagrama de Gantt - Calendario del Proyecto

Fechas: 08/02/2024 – 20/03/2024

El comienzo del desarrollo parte del análisis de requisitos realizado durante el periodo de prácticas en el departamento del SOUCAN, a partir de este trabajo se cuenta con un documento el cual figura. Este trabajo produjo un documento que recoge todos los requisitos funcionales requeridos para conseguir una funcionalidad completa, los cuales se determinaron a partir de reuniones con las personas encargadas de la gestión del servicio.

Fechas: 21/03/24 – 25/03/24

A través de varias reuniones con el responsable del departamento del Área de Servicios Web de la Universidad de Cantabria, se identificaron los requisitos no funcionales necesarios para el desarrollo del proyecto. En estas reuniones se definieron los atributos de calidad y las restricciones que debe cumplir el servicio, así como las tecnologías que permitirán abordar estos aspectos.

Sección del índice: software utilizado + requisitos funcionales y no funcionales

Fechas: 26/03/24-27/03/24

Tras realizar el análisis de los requisitos, se procede a la fase de modelado del dominio. Esta fase tiene como objetivo representar todos los objetos y relaciones necesarios para implementar la funcionalidad deseada y cumplir con los requisitos establecidos.

Sección del índice: desarrollo, diseño e implementación.

Fechas: 28/03/24-01/04/24

Con base en el modelo de dominio previamente desarrollado, se procede a diseñar el almacenamiento de la información en la base de datos. Este modelado relacional se diseña con el fin de preservar la semántica y distribución de los objetos junto con la manera en la que se encuentran relacionados, con el objetivo de almacenar los datos de manera que se pueda acceder y manipular de forma lógica y eficiente.

Sección del índice: desarrollo, diseño e implementación.

Fechas: 01/04/24-15/04/24

Después de diseñar las entidades necesarias para el desarrollo, se procede a la conversión del modelo relacional a tablas. Para ello, se utiliza la herramienta "Microsoft SQL Server", que proporciona los recursos necesarios para almacenar todos los objetos manipulados en un servidor. Este proceso se divide en varias subetapas:

1. Creación y paso a tablas: Se utiliza el modelo relacional previo para definir la distribución de las tablas, así como el tipo de datos y las restricciones de sus campos.
2. Generación de objetos para testeo posterior: Tras la creación de las tablas, se generan una serie de objetos "testeables" para que, durante el desarrollo de la API, se pueda probar cómo reaccionan al utilizar los distintos procedimientos almacenados.
3. Desarrollo de procedimientos almacenados: Con los objetos de la aplicación ya disponibles, se procede a la creación de los procedimientos necesarios para que el servicio funcione correctamente y realice las operaciones de lectura y escritura definidas en el análisis de requisitos. Además, se verifica el correcto funcionamiento de estos procedimientos utilizando los elementos de prueba previamente generados.

Sección del índice: desarrollo, diseño e implementación.

Fechas: 15/04/24-16/04/24

En esta etapa, se inicia la implementación de la API (Interfaz de Programación de Aplicaciones) utilizando "Visual Studio", que facilita la subdivisión en capas mediante el uso del modelo MVC (Modelo-Vista-Controlador). Dentro de esta estructura, se establece la conexión con el servidor de base de datos para realizar el mapeo de los objetos y los procedimientos almacenados.

Sección del índice: desarrollo, diseño e implementación.

Fechas: 16/04/24-22/04/24

Se desarrolla la capa Modelo junto con sus subdivisiones (Capa Entidad, Capa Datos y Capa Negocio). Además, se crea un ejecutable de prueba para comprobar el correcto mapeo de los elementos. A continuación, se detalla la utilidad de las distintas partes:

1. Capa Entidad: En esta capa se define la estructura de los distintos objetos de la API, permitiendo representar los objetos de la base de datos y trabajar con ellos a través del servicio.
2. Capa Datos: Su función es mapear todas las operaciones que permiten leer y escribir los elementos de la base de datos. El desarrollo de esta sección es laborioso, ya que es necesario asegurar que todos los elementos almacenados se relacionen adecuadamente con los definidos en la Capa Entidad.
3. Capa Negocio: Se encarga de filtrar los métodos provenientes de la Capa de Datos, verificando que los parámetros sean correctos y permitiendo que se puede utilizar una operación de la Capa de Datos como varias suboperaciones en la Capa de Negocio. Además, en esta capa se crean las operaciones necesarias para complementar la funcionalidad del servicio.

Sección del índice: desarrollo, diseño e implementación

Fechas: 22/04/24-29/04/24

En este periodo se completa la Capa Controlador y se estructura la Capa Vista.

1. Capa Controlador: Se encarga de conectar la manipulación de los datos a través de las vistas de la web con las operaciones implementadas en la Capa de Negocio. También establece cómo se relacionan y comunican las distintas vistas de la web.
2. Capa Vista: Únicamente se implementa la estructura y la relación entre las distintas vistas, utilizando el mock de interfaces desarrollado en el periodo de prácticas. Posteriormente se diseñará e implementará las mismas.

Sección del índice: desarrollo, diseño e implementación

Fechas: 29/04/24-08/05/24

Entre estas fechas, una vez finalizado el desarrollo del Back-End de la aplicación, se procede a trabajar en el Front-End, es decir, a estructurar el apartado visual disponible en la interfaz del usuario. Para esta etapa, se utilizan las vistas generadas previamente junto con el controlador. Esta tarea es bastante compleja, ya que requiere un estudio previo sobre cómo implementar y combinar HTML con el framework seleccionado (.NET Framework).

Sección del índice: desarrollo, diseño e implementación

Fechas: 08/05/24-14/05/24

Finalmente, tras completar el desarrollo del servicio, se implementa la sección de Access para su uso local por los empleados encargados de la gestión del SOUCAN. Antes de esta implementación, se realizan dos reuniones con el encargado del Área de Servicios Web de la universidad para comprender el uso de este entorno de desarrollo. Posteriormente, se implementan todas las funcionalidades solicitadas y se verifica su correcto funcionamiento mediante pruebas de manipulación de datos, asegurando su persistencia en la base de datos.

Sección del índice: Desarrollo, diseño e implementación

# SOFTWARE UTILIZADO

En el desarrollo de esta aplicación se han utilizado diversas herramientas y plataformas de software esenciales para alcanzar los objetivos del proyecto. Se detallan las tecnologías seleccionadas, explicando su elección y cómo cada una contribuye a la funcionalidad y calidad de la aplicación final. Esta sección se encuentra subdividida de acuerdo con las distintas partes de la aplicación:

1. Diseño de la aplicación:
   1. Umlet: Permite realizar un diseño del modelo de dominio de una manera sencilla e intuitiva, contando con una plantilla con todos los elementos necesarios para esta tarea.
2. Base de datos:
   1. Microsoft SQL Server: se utiliza para gestionar la base de datos de la aplicación. Se emplea para la creación de tablas, la inserción de datos de prueba y el desarrollo de procedimientos almacenados. Estas funciones son esenciales para estructurar, organizar y establecer restricciones para los datos, garantizar la integridad de la información y optimizar las operaciones de consulta y manipulación de datos dentro del sistema.
3. API (Interfaz de Programación de Aplicaciones):
   1. Microsoft Visual Studio: empleado para la realización de la lógica de la aplicación, junto con el mapeado e interfaces. Se aplica el modelo MVC, el cual permite que se distribuya el desarrollo de la aplicación en capas, permitiendo así una implementación entendible y apta para cambios futuros.
   2. .NET Framework: framework que se emplea para desarrollar la aplicación, dando una plataforma fiable y versátil para la construcción de aplicaciones seguras y eficientes. En este proyecto, facilita la integración de distintas funcionalidades, la gestión de la lógica de negocio y la creación de una interfaz de usuario.
4. Trabajo local con base de datos:
   1. Microsoft Access: se utiliza para gestionar y almacenar datos a nivel local. En este proyecto, Access permite realizar tareas de gestión del SOUCAN de manera eficiente, proporcionando una interfaz para la manipulación de datos y garantizando su persistencia en la base de datos.
5. Almacenamiento en Nube:
   1. GitHub: para el almacenamiento distribuido de los documentos y archivos resultantes del desarrolla se utiliza esta plataforma, la cual permite establecer el proyecto como una fuente de software libre y aporta la capacidad de realizar un control de versiones durante la implementación.

# REQUISITOS FUNCIONALES Y NO FUNCIONALES

## Requisitos Funcionales:

Este apartado no aplica al desarrollo del TFG, ya que los requisitos funcionales necesarios para el correcto desarrollo de la aplicación se recogieron durante el periodo de prácticas previo. Durante estas prácticas, se identificaron todas las necesidades funcionales del departamento del SOUCAN para esta tarea. Por lo tanto, los detalles relevantes se pueden encontrar en el informe de prácticas curriculares.

## Requisitos no funcionales:

1. Eficiencia: La aplicación debe operar de manera eficiente, optimizando el uso de recursos del sistema para asegurar un rendimiento adecuado, tanto en la manipulación de datos en Microsoft Access como en las interacciones con la API y SQL Server.

2. Escalabilidad: La estructura de la aplicación debe permitir futuras ampliaciones y mejoras sin impedir su funcionalidad actual. Esto incluye la capacidad de añadir nuevas tablas, procedimientos almacenados y funcionalidades en el servicio. De hecho, debe estar diseñada de tal forma que se pueda implementar en un futuro el uso del servicio de manera remota para las tareas aplicadas a través de Access.

3. Seguridad: Se deben implementar medidas de seguridad robustas para proteger los datos almacenados en SQL Server y accesibles a través de Microsoft Access. Esto incluye la autenticación a través de registro de usuarios y la encriptación de datos sensibles, como contraseñas.

4. Usabilidad: La interfaz de usuario desarrollada en el .NET Framework debe ser intuitiva y fácil de usar, facilitando la gestión por parte de los centros educativos. La integración con Microsoft Access debe permitir una manipulación de datos fluida y eficiente.

5. Mantenibilidad: El código y la estructura de la aplicación deben ser fácilmente mantenibles. Esto incluye una documentación clara y comprensible gracias al informe del TFG, así como la adopción de buenas prácticas de programación que faciliten futuras actualizaciones y correcciones gracias a el uso de interfaces.

6. Compatibilidad: La aplicación debe ser compatible con las versiones de software y hardware utilizadas por el departamento del SOUCAN, garantizando un funcionamiento sin problemas en el entorno operativo existente. Esto quedó concretado junto con el Departamento de Área Web de la universidad, el cual conoce los requisitos necesarios para que se cumpla esto en cada departamento.

7. Fiabilidad: La aplicación debe ser confiable, asegurando que todas las funcionalidades, desde la gestión de datos en Microsoft Access hasta las operaciones de la API a través de los procedimientos almacenados de SQL Server, funcionen correctamente y sin errores.

# CASOS DE USO

Para realizar la explicación de los casos de uso en el contexto del desarrollo del proyecto se realiza una estructuración en la que cada caso de uso describe las interacciones entre los usuarios y el sistema, detallando los actores, las condiciones, los flujos de operación y los requisitos específicos, proporcionando una visión clara del funcionamiento del sistema y su relación con los requisitos del proyecto.

1. Inicio Sesión Centro
   1. Descripción: Permite a un centro registrado por el SOUCAN iniciar sesión para tener acceso al servicio.
   2. Actores:
      1. Centro Educativo.
      2. Sistema de Autenticación.
   3. Precondiciones:
      1. El centro ha de estar previamente registrado.
   4. Flujo Principal:
      1. El centro accede a la pantalla de inicio de sesión.
      2. El centro introduce su correo y contraseña.
      3. El sistema verifica la existencia del usuario.
      4. El sistema da acceso al centro para el uso del servicio.
   5. Flujos Alternativos:

3a. El sistema no encuentra el centro con dicho correo y contraseña y muestra mensaje de error

3b. El centro introduce datos válidos y vuelve al paso 3.

* 1. Postcondiciones:
     1. Se debe de conceder acceso al centro al servicio
     2. Únicamente se debe de mostrar los datos asociados al centro

1. Cambio de contraseña Centro:
   1. Descripción: Permite a un centro realizar el cambio de contraseña de su cuenta personal dentro del servicio.
   2. Actores:
      1. Centro Educativo.
      2. Sistema de Autenticación.
   3. Precondiciones:
      1. El centro debe de haber iniciado sesión previamente.
   4. Flujo Principal:
      1. El centro pulsa el botón “Cambiar Contraseña”.
      2. El centro introduce la contraseña previa, la nueva contraseña y la nueva contraseña repetida.
      3. El sistema verifica la contraseña anterior y cambia a la nueva.
      4. El sistema redirige al usuario al inicio de sesión.
   5. Flujos Alternativos:

3a. El sistema detecta que la contraseña previa no es correcta y muestra mensaje de error.

3b. El centro introduce la contraseña correcta y vuelve al paso 3.

* 1. Postcondiciones:
     1. Se debe de redirigir al centro al inicio de sesión.
     2. El centro debe poder acceder al servicio con su nueva contraseña.
     3. El centro no debe poder acceder al servicio con la antigua contraseña.

1. Editar información Centro:
   1. Descripción: Permite a un centro realizar el cambio de los datos asociados a su cuenta personal.
   2. Actores:
      1. Centro Educativo.
      2. Sistema de Gestión de Usuarios.
   3. Precondiciones:
      1. El centro debe de haber iniciado sesión previamente.
   4. Flujo Principal:
      1. El centro pulsa el botón “Editar Información Centro”.
      2. El centro modifica los campos que se le permite editar acerca de su cuenta.
      3. El usuario pulsa el botón “Guardar Cambios”.
      4. El sistema verifica la correctitud de las entradas y cambia los datos del centro.
      5. El sistema redirige al centro al menú principal
   5. Flujos Alternativos:

4a. El sistema detecta que los números de teléfono introducidos no tienen el formato correcto y el sistema muestra el mensaje de error con los campos incorrectos.

4b. El centro introduce los números de teléfono correctamente y vuelve al paso 3.

* 1. Postcondiciones:
     1. Se debe de redirigir a la página principal.
     2. Los cambios de información acerca del centro deben persistir.

1. Registrar Estudiante Centro:
   1. Descipción: Permite a un centro registrar a estudiantes con sus debidas adaptaciones.
   2. Actores:
      1. Centro Educativo.
      2. Sistema de Gestión de Estudiantes.
   3. Precondiciones:
      1. El centro debe de haber iniciado sesión previamente.
      2. La fecha actual debe de encontrarse dentro del plazo de registro activo.
   4. Flujo Principal:
      1. El centro pulsa el botón “Registrar Estudiante”.
      2. El centro introduce los datos del estudiante, junto con sus asignaturas previstas para la EBAU, las adaptaciones necesarias y los documentos solicitados.
      3. El centro pulsa el botón “Registrar”.
      4. El sistema verifica la correctitud de las entradas y registra al estudiante.
      5. El sistema redirige al centro a el listado de estudiantes registrados.
   5. Flujos Alternativos:

3a. El sistema detecta que los datos introducidos son incorrectos o insuficientes y muestra un mensaje de error con todos los cambios necesarios

3b. El centro introduce los cambios necesarios y vuelve al paso 3.

* 1. Postcondiciones:
     1. Se debe redirigir al listado de estudiantes del centro.
     2. Se debe establecer el estado del estudiante como “Pendiente”, lo cual indica que el SOUCAN no ha comprobado la correctitud de los datos todavía.
     3. Se debe de permitir posteriormente al centro a acceder al estudiante registrado.
     4. El estudiante debe de tener asociado el plazo de registro automáticamente

1. Revisión datos estudiante:
   1. Descripción: Permite a un centro visualizar el estado de las solicitudes de un estudiante.
   2. Actores;
      1. Centro Educativo
   3. Precondiciones:
      1. El centro debe de haber iniciado sesión previamente.
      2. El estudiante debe de estar registrado previamente (caso de uso: Registrar Estudiante Centro).
   4. Flujo Principal:
      1. El centro pulsa el botón “Ver Estudiantes Registrados”.
      2. El centro pulsa el botón “Ver” de la fila asociada con el estudiante que desea revisar.
      3. El centro comprueba el estado general del estudiante junto con el estado de los documentos y adaptaciones registrados.
   5. Postcondiciones:
      1. En el caso de tener apartados en estado “Rechazado”, el centro debe de modificar la solicitud del estudiante para una posterior revisión.
2. Edición de Estudiante Centro:
   1. Descripción: Permite a un centro editar la solicitud de un estudiante.
   2. Actores:
      1. Centro Educativo.
      2. Sistema de Gestión de Estudiantes
   3. Precondiciones:
      1. El centro debe de haber iniciado sesión previamente.
      2. El estudiante no debe estar ya validado, ya que, en ese caso no se permite realizar la acción.
      3. La fecha actual debe de encontrarse dentro del plazo de registro activo.
   4. Flujo Principal:
      1. El centro pulsa el botón “Ver Estudiantes Registrados”.
      2. El centro pulsa el botón “Editar” de la fila asociada con el estudiante que desea revisar.
      3. El centro modifica los datos de la solicitud del estudiante que se encuentren pendientes o rechazados y que se permitan editar.
      4. El centro pulsa el botón “Guardar cambios”.
      5. El sistema verifica la correctitud de los campos modificados y edita el estudiante.
      6. El sistema redirige al centro a el listado de estudiantes registrados.
   5. Postcondiciones:
      1. Se debe redirigir al listado de estudiantes del centro.
      2. Se debe establecer el estado del estudiante como “Pendiente”, lo cual indica que el SOUCAN no ha comprobado la correctitud de los datos modificados todavía.
      3. El cambio de los datos modificados del estudiante debe persistir.
3. Edición Plazo de Registro:
   1. Descripción: Permite al SOUCAN establecer el periodo de registro activo.
   2. Actores:
      1. SOUCAN.
      2. Sistema de Gestión de Base de Datos (Microsoft Access).
   3. Precondiciones:
      1. Debe de existir un plazo de registro activo previo que se ha de desactivar.
   4. Flujo Principal:
      1. El SOUCAN accede a la aplicación de Microsoft Access.
      2. El SOUCAN pulsa el botón “Plazos Registro”
      3. El SOUCAN introduce un nuevo plazo de registro
      4. El sistema registra el plazo de registro activo en la base de datos.
   5. Postcondiciones:
      1. El nuevo plazo ha de establecerse como activo automáticamente.
      2. El nuevo plazo ha de estar disponible en el servicio.
4. Modificación de Formulario
   1. Descripción: Permite al SOUCAN realizar cambios de los datos disponibles a seleccionar en el formulario de registro de estudiantes.
   2. Actores:
      1. SOUCAN.
      2. Sistema de Gestión de Base de Datos (Microsoft Access).
   3. Precondiciones:
      1. Debe existir la opción de cambiar la lista de adaptaciones disponibles.
      2. Debe existir la opción de cambiar la lista de diagnósticos disponibles.
      3. Debe existir la opción de cambiar la lista de sedes disponibles.
      4. Debe existir la opción de cambiar la lista de asignaturas disponibles.
      5. Debe existir la opción de modificar las asociaciones entre diagnósticos y adaptaciones
   4. Flujo Principal:
      1. El SOUCAN accede a la aplicación de Microsoft Access.
      2. El SOUCAN pulsa el botón “Edición Formularios”.
      3. El SOUCAN modifica los datos del formulario que desee.
      4. El sistema guarda los cambios realizados en la base de datos.
   5. Postcondiciones:
      1. El formulario cambiado ha de estar disponibles en el servicio.
      2. Los campos que hayan sido eliminados del formulario no deben desaparecer si no marcarse como inactivos.
5. Acceder y revisar solicitud estudiante:
   1. Descripción: Permite al SOUCAN validar o rechazar solicitudes de estudiantes.
   2. Actores:
      1. SOUCAN.
      2. Sistema de Gestión de Base de Datos (Microsoft Access).
   3. Precondiciones:
      1. Los centros han de registrar a los estudiantes junto con sus solicitudes previamente.
   4. Flujo Principal:
      1. El SOUCAN accede a la aplicación Microsoft Access.
      2. El SOUCAN pulsa el botón “Información Centros Educativos”.
      3. El SOUCAN se mueve entre registros hasta llegar al centro deseado.
      4. El SOUCAN pulsa la ventana Estudiantes Centro.
      5. El SOUCAN se mueve entre registros hasta llegar al estudiante deseado.
      6. El SOUCAN se mueve entre las subpestañas cambiando el campo “validado” de pendiente a validado o rechazado.
      7. El sistema guarda los cambios realizados en la base de datos
   5. Postcondiciones:
      1. Los centros han de poder visualizar los cambios de estado de los estudiantes desde el servicio.
      2. Si un estudiante está completamente validado, no se podrá modificar posteriormente.

# ARQUITECTURA

Esta sección tiene como objetivo proporcionar una visión clara y detallada de la arquitectura de la aplicación, destacando sus componentes fundamentales, sus interacciones y las tecnologías utilizadas. A través de diagramas y descripciones detalladas, se establecerá una base sólida para la implementación del sistema, garantizando que se cumplan los principios de escalabilidad, mantenibilidad y seguridad.

## Descripción de Componentes y Conexiones:

### Base de datos:

La base de datos del sistema ha sido desarrollada utilizando SQL Server. Esta base de datos se compone de dos secciones principales: las tablas que almacenan los datos y los procedimientos almacenados que gestionan toda la lógica de negocio.

#### TABLAS

* Descripción:
  + Las tablas en SQL Server se utilizan para almacenar los datos de la aplicación de manera estructurada. Cada tabla está diseñada con columnas que representan los atributos de las entidades que forman parte del modelo relacional.
* Principales Entidades:
  + Usuario: Almacena la información acerca de los usuarios registrados en el sistema. Está diseñada para ser ampliable y compatible con los usuarios del sistema SOUCAN. Actualmente, se extiende a "CentroEducativo", una entidad que almacena toda la información relacionada con los centros educativos registrados en el sistema. Además, se utilizan varias tablas auxiliares para la definición de atributos específicos:
    - Dirección: Contiene la dirección en la que se ubica el centro en forma de texto plano. Está diseñado de esta manera por si en algún momento se desea seccionar la misma en distintos apartados (Código Postal, Localidad, Calle, etc.).
    - Municipio: Almacena todos los municipios existentes en Cantabria, permitiendo a los centros educativos seleccionar su ubicación de entre los municipios disponibles. Esta tabla garantiza la normalización de los datos de ubicación.
    - Sede: Registra las distintas sedes disponibles para la realización de la prueba de EBAU. Incluye la capacidad de activar o desactivar sedes, lo cual permite controlar su visibilidad en los formularios del sistema. Esta funcionalidad es esencial para la gestión dinámica de las sedes.
  + Estudiante: Recoge la información acerca de los estudiantes registrados a través del servicio por cada centro educativo, haciendo referencia al centro educativo al que pertenece cada estudiante. Además, tiene una asociación con una serie de tablas intermedias que permiten almacenar los datos relacionados con la solicitud realizada al SOUCAN, mencionadas a continuación, junto con una tabla auxiliar:
    - Apunte: Permite al SOUCAN recoger información en forma de texto plano acerca de los documentos resultantes de las reuniones que se realizan con los orientadores de los centros sobre el estudiante en concreto.
  + Asignatura: Describe las asignaturas disponibles en el formulario para que los estudiantes puedan indicar su previsión de presentación, además de las asignaturas a las que se matriculan finalmente. Cuenta con la capacidad de activar o desactivar asignaturas, permitiendo ajustarse a los cambios que puedan ocurrir en cada convocatoria.
  + Documento: Establece qué documentos deben aportar los estudiantes en los formularios, con el fin de justificar la solicitud de adaptaciones. Permite almacenar la ruta para que el SOUCAN conozca dónde ubicar los documentos adjuntados en el registro.
  + Diagnóstico: Permite definir qué diagnósticos están disponibles para la solicitud. Además, cuenta con la posibilidad de activarlos o desactivarlos para mostrarlos o no en el formulario a rellenar por los centros.
  + Adaptación: Aporta la capacidad de establecer qué adaptaciones están disponibles para cada diagnóstico existente mediante una tabla intermedia de asociación. La tabla de adaptaciones permite explicar el razonamiento y la descripción de cómo cada adaptación servirá durante el examen, y especificar si es excepcional o no, junto con su descripción excepcional. Además, cuenta con la posibilidad de activarlos o desactivarlos para mostrarlos o no en el formulario a rellenar por los centros.
  + Plazo Registro: Tiene la funcionalidad de describir los plazos que se encuentran activos para el registro de estudiantes. Además, permite establecer en cada estudiante un campo que describe el plazo en el que se ha inscrito.

Diagrama, Esquemático

Descripción generada automáticamente

Ilustración 2: Diagrama de Tablas (SQL Server)

#### PROCEDIMIENTOS ALMACENADOS:

* Descripción: Los procedimientos almacenados en SQL Server gestionan la lógica de negocio del sistema, encapsulando las operaciones complejas que mejoran la eficiencia, seguridad y mantenibilidad del código. Todas las operaciones comprueban la correctitud de las salidas para evitar errores más adelante.
* Procedimientos definidos:
  + Lectura: Esta sección engloba todas las operaciones de lectura que se realizan sobre la base de datos, es decir, las operaciones que recuperan datos de las tablas almacenadas en el sistema:
    - sp\_obtenPlazoRegistroActivo: recupera el único plazo que se encuentra activo.
    - sp\_obtenMunicipioDireccion: obtiene los datos del municipio que se encuentra asociado a una dirección.
    - sp\_obtenDireccionCentro: obtiene los datos de la dirección que se encuentra asociada a un centro educativo.
    - sp\_obtenSedeCentro: obtiene los datos de la sede que se encuentra asociada a un centro educativo.
    - sp\_obtenAdaptacionDiagnostico: ontiene la adaptacion que se encuentra asociada a un diagnóstico.
    - sp\_listaDiagnosticosEstudiante: obtiene los diagnósticos que se encuentran asociados a un estudiante.
    - sp\_listaAdaptacionesDiagnosticoEstudiante: ontiene las adaptaciones que se encuentran asociadas a un diagnóstico que a su vez se encuentran asociadas a un estudiante.
    - sp\_obtenAsignaturasPrevistasEstudiante: obtiene las asignaturas previstas que se encuentran asociadas a un estudiante.
    - sp\_obtenAsignaturasMatriculadasEstudiante: obtiene las asignaturas matriculadas que se encuentran asociadas a un estudiante.
    - sp\_listaApuntesEstudiante: obtiene los apuntes que se encuentran asociados a un estudiante.
    - sp\_obtenCentro: obtiene los datos de un centro en concreto.
    - sp\_obtenEstudianteCentro: obtiene los datos de un estudiante asociado a un centro.
    - sp\_listaAdaptacionesDiagnostico: obtiene las adaptaciones que se encuentran asociadas a un diagnóstico.
    - sp\_obtenDiagnostico: obtiene los datos de un diagnóstico.
    - sp\_obtenAdaptacion: Obtiene los datos de una adaptación.
    - sp\_listaEstudiantesCentro: obtiene los estudiantes que se encuentran asociados a un centro educativo.
    - sp\_listaSedes: obtiene todas las sedes disponibles.
    - sp\_listaMunicipios: obtiene todos los municipios disponibles.
    - sp\_listaDiagnosticos: obtiene todos los diagnósticos disponibles.
    - sp\_listaAdaptaciones: obtiene todas las adaptaciones disponibles.
    - sp\_listaAsignaturas: obtiene todas las asignaturas disponibles.
    - sp\_listaDocumentos: obtiene todos los documentos que se encuentran disponibles.
    - sp\_listaDocumentosEstudiante. obtiene todos los documentos que se encuentran asociados a un estudiante.
    - sp\_listaDiagnosticosEstudiante: obtiene todos los diagnósticos que se encuentran asociados a un estudiante.
  + Inserción: Esta sección engloba parte las operaciones de escritura que se realizan sobre la base de datos, en concreto las operaciones que insertan datos en las tablas existentes:
    - sp\_registraCentroEducativo: registra un nuevo centro educativo en la base de datos junto con sus datos pasados como parámetros
    - sp\_registraEstudiante: registra un nuevo estudiante en la base de datos junto con sus datos pasados como parámetros
    - sp\_registraAsignaturaPrevistaEstudiante: registra una asignatura prevista dentro de las disponibles asociándolo a un estudiante junto con las fases que se pasan como parámetros
    - sp\_registraDocumentoEstudiante: registra un documento asociándolo a un estudiante junto con la ruta de acceso al mismo pasada como parámetro
    - sp\_registraAdaptacionDiagnosticoEstudiante: registra una adaptación junto con el diagnóstico al que se encuentra asociada en un estudiante, permitiendo añadir un campo de observaciones pasado como parámetro.
  + Eliminación: Esta sección engloba todas las operaciones de eliminación que se realizan sobre la base de datos, es decir, las operaciones eliminan filas de las tablas existentes:
    - sp\_eliminaDiagnosticoEstudiante: Elimina un diagnóstico de un estudiante junto con todas las adaptaciones que se encontraban asociadas al mismo.
    - sp\_eliminaAsignaturasPrevistasEstudiante: Elimina todas las asignaturas previstas que se encontraban asociadas a un estudiante.
    - sp\_eliminaDocumentosEstudiante: Elimina todos los documentos que se encontraban asociados a un estudiante, incluyendo las rutas epsecificadas.
  + Modificación: Esta sección engloba la otra parte de las operaciones de escritura que se realizan sobre la base de datos, es decir, cambian valores de los atributos de las filas ya existentes en las tablas existentes:
    - sp\_modificaDatosCentro: Modifica los campos que se pasan como parámetros de los datos de un centro.
    - sp\_modificaDatosEstudiante: Modifica los campos que se pasan como parámetros de los datos de un estudiante.
  + Auxiliares: Esta sección concreta cuenta con funcionalidades específicas que requiere la implementación del servicio:
    - sp\_cambiaContrasenha: Modifica el campo contraseña de un usuario, comprobando que la contraseña aportada original es correcta.
    - sp\_validarCentroEducativo: Se encarga de comprobar las credenciales de correo y contraseña de un usuario.
  + No Aplicados: esta sección se basa en varios procedimientos que fueron implementados previamente para el desarrollo del servicio pero no han sido requeridos para lograr el alcance especificado. Estos se mantienen debido a que, para implementaciones futuras podrán ser de utilidad, debido a que incluyen parte de las funciones requeridas para ampliar el servicio web a que albergue el uso por parte del SOUCAN:
    - sp\_obtenApunteEstudiante: obtiene todos los apuntes que se encuentran asociados a un estudiante.
    - sp\_gestionaEstadoDiagnostico: permite cambiar el atributo “activo” a un diagnóstico.
    - sp\_gestionaEstadoAdaptacion: permite cambiar el atributo “activo” a una adaptación.
    - sp\_gestionaEstadoAsignatura: permite cambiar el atributo “activo” a una asignatura.
    - sp\_gestionaEstadoSede: permite cambiar el atributo “activo” a una sede.
    - sp\_asociaAdaptacionADiagnostico: permite establecer la relación entre un diagnóstico y una adaptación a través de una tabla intermedia.
    - sp\_anhadeDiagnostico: registra un nuevo diagnóstico junto con sus atributos.
    - sp\_anhadeAdaptacion: registra una nueva adaptación junto con sus atributos.
    - sp\_anhadeAsigantura: registra una nueva asignatura junto con sus atributos.
    - sp\_anhadeApunteEstudiante: permite registrar un nuevo apunte para un estudiante en concreto.
    - sp\_muestraInfoEstudiante: recupera la información acerca de un estudiante
    - sp\_registraAsignaturaMatriculadaEstudiante: permite registrar una asignatura junto con sus fases a un estudiante concreto.
    - sp\_listaAsignaturasPrevistasEstudianteF1: obtiene todas las asignaturas previstas de la fase 1 asociadas a un estudiante.
    - sp\_listaAsignaturasPrevistasEstudianteF2: obtiene todas las asignaturas previstas de la fase 2 asociadas a un estudiante.

### Back-End:

El back-end de la aplicación ha sido desarrollado en C# utilizando el .NET Framework en el entorno de desarrollo Visual Studio. Este componente se encarga de manejar la lógica de negocio, aplicar las operaciones del servidor y gestionar la comunicación con la base de datos.

Utilizando el .NET Framework, el back-end se beneficia de su robustez, escalabilidad y rendimiento eficiente. Este framework proporciona un amplio conjunto de herramientas y librerías que simplifican el desarrollo de aplicaciones. La elección de C# y .NET Framework permite aprovechar características avanzadas del desarrollo de servicios web.

Además, el back-end se integra con SQL Server a través de una conexión para gestionar los datos de manera eficiente y segura a través del mapeado de los procedimientos almacenados mencionados previamente. Esta integración asegura que la información almacenada sea consistente, disponible y accesible para las funcionalidades de la aplicación.

El uso de Visual Studio como entorno de desarrollo proporciona un entorno integrado y robusto para la escritura, depuración y prueba del código, contribuyendo notablemente a una mejor productividad y calidad en el desarrollo.

A continuación, se presenta el modelo de dominio empleado en el desarrollo:

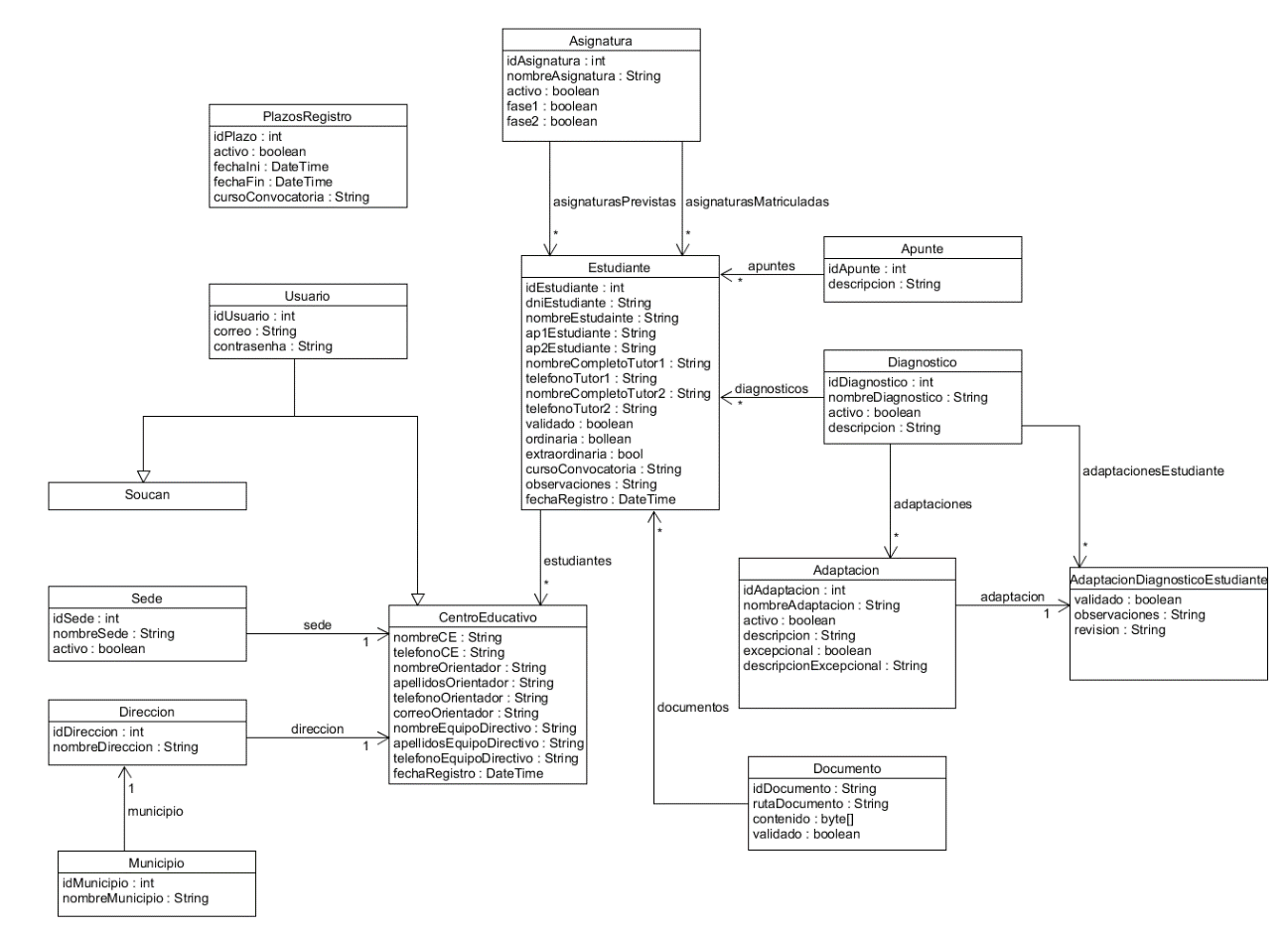


Ilustración 3: Modelo de Dominio

Front-End:

El front-end de la aplicación ha sido desarrollado utilizando .NET Framework en combinación con Bootstrap para crear una interfaz de usuario interactiva y responsiva. La estructura del front-end integra varias vistas interconectadas entre sí, proporcionando una experiencia de usuario coherente y fluida.

Las vistas están diseñadas utilizando HTML 5 y se apoyan en Bootstrap para asegurar un diseño responsivo y comprensible. Bootstrap proporciona estilos y componentes predefinidos, lo que acelera el desarrollo y garantiza un uso consistente en toda la aplicación.

Para manejar la lógica de presentación y la interacción entre las vistas y los datos, se utilizan ViewModels. Estos permiten separar la lógica de la interfaz de usuario de la lógica de negocio, facilitando la gestión de los datos sobre las vistas. Esta arquitectura mejora la mantenibilidad y escalabilidad del código, permitiendo una mejor organización y reutilización de componentes.

El uso de .NET Framework en el front-end permite una asociación con el back-end, asegurando una comunicación eficiente y segura entre el cliente y el servidor. Visual Studio sirve como el entorno de desarrollo integrado, proporcionando herramientas avanzadas para el diseño, depuración y prueba del front-end, lo que contribuye a una mayor productividad y calidad en el desarrollo de la interfaz de usuario.

## Patrón de Diseño:

La arquitectura de la aplicación sigue el patrón de diseño MVC (Model-View-Controller), que divide la aplicación en tres capas principales: Modelo, Controlador y Vista. Esta separación facilita la gestión del código, mejora la mantenibilidad y permite un desarrollo más organizado.

Modelo:

* Capa Entidad: Define las entidades que representan los datos del dominio. Estas entidades son clases que encapsulan las propiedades y comportamientos de los objetos que se enlazan con las tablas de la base de datos con la API.
* Capa Datos: Maneja la comunicación con la base de datos. Incluye el acceso a los procedimientos almacenados que encapsulan toda la lógica de negocio permitiendo así aplicarlo al servicio a través del mapeado de los mismos.
* Capa Negocio: Contiene la comprobación de lógica de negocio de la aplicación. Esta capa procesa los parámetros introducidos para comprobar su correctitud y dar una respuesta correcta o mostrar un mensaje de error al emplearlo para aportar información al controlador.
* Controlador: Actúa como intermediario entre el Modelo y la Vista. Los controladores reciben las entradas del usuario a través de la Vista, procesan estas entradas empleando los métodos de la Capa Negocio, y finalmente actualizan la Vista con los datos apropiados. Esta capa asegura que la lógica de la aplicación esté separada de la interfaz de usuario.
* Vista: Responsable de la presentación de los datos. Las vistas generan la interfaz de usuario y presentan los datos al usuario final. Utiliza HTML, CSS y Bootstrap para asegurar una presentación atractiva y responsiva. Las vistas se actualizan basándose en los datos proporcionados por los controladores.

En conjunto, estas tres capas trabajan de manera cohesiva para proporcionar una aplicación bien estructurada y fácil de mantener. La implementación del patrón MVC con .NET Framework asegura una clara separación de responsabilidades y facilita la colaboración en el desarrollo de la aplicación.

## Escalabilidad y Seguridad:

La aplicación está diseñada teniendo en cuenta tanto la escalabilidad como la seguridad para garantizar un rendimiento óptimo y la protección de los datos en un entorno en constante evolución:

1. Escalabilidad: La aplicación ha sido diseñada de manera que permite ser fácilmente ampliable sin necesidad de modificar las implementaciones previas. Esta capacidad de escalabilidad se logra a través de prácticas que aseguran que el sistema pueda crecer y adaptarse a las nuevas necesidades sin comprometer la estabilidad y funcionalidad actual.
   1. Arquitectura Modular:
      1. Modularidad: La aplicación se encuentra dividida en módulos independientes, cada uno con una responsabilidad específica. Esto permite añadir nuevas funcionalidades simplemente creando nuevos módulos o extendiendo los existentes sin afectar otras partes del sistema.
      2. Interfaz Bien Definida: Los módulos se comunican entre sí mediante interfaces bien definidas y documentadas, lo que facilita la integración de nuevos componentes. Esto aplica en concreto a la capa de negocio que conecta el Back-End con el Front-End.
   2. Principio de Abierto/Cerrado:
      1. Abierto para Extensión, Cerrado para Modificación: La arquitectura sigue el principio de diseño SOLID de estar abierta para la extensión, pero cerrada para la modificación. Esto significa que se pueden añadir nuevas funcionalidades mediante la extensión de clases o la adición de nuevos módulos sin la necesidad de alterar el código existente.
2. Seguridad: Para garantizar la seguridad de la aplicación, se han integrado varias medidas clave que protegen tanto los datos como el acceso a la información.
   1. Autenticación:
      1. Sistema de Login: La aplicación requiere que los usuarios se autentiquen mediante un sistema de login antes de poder acceder a sus funcionalidades. Este proceso asegura que solo los usuarios autorizados puedan entrar al sistema.
   2. Autorización:
      1. Gestión de permisos: Los permisos se gestionan cuidadosamente para asegurar que cada usuario únicamente pueda realizar acciones y acceder a los datos que le pertenecen. Esto evita el acceso no autorizado a información sensible.
   3. Prevención de Inyecciones SQL:
      1. Consultas Parametrizadas: Todas las interacciones con la base de datos se realizan utilizando consultas parametrizadas. Esto asegura que los datos ingresados por los usuarios se manejen de manera segura, evitando inyecciones SQL.
   4. Encriptación de Datos:
      1. Datos en Reposo: Los datos sensibles almacenados en la base de datos, como las contraseñas de los usuarios, se encriptan. Esto asegura que, incluso si la base de datos es comprometida, los datos permanezcan protegidos. De esta manera, no se puede acceder a la información de un usuario sin su contraseña, garantizando la seguridad de la información almacenada.

## Herramientas y Recursos:

aDF

# DESARROLLO, DISEÑO E IMPLEMENTACIÓN

## Diseño de Base de Datos

### Modelado de ER (Modelo Relacional)

### Paso a tablas

## Diseño e implementación de procedimientos almacenados (Lógica de Negocio)

## Diseño de Modelo de Dominio

## Desarrollo de API (Interfaz de Programación de Aplicaciones)

### Implementación de Modelo de Dominio

### Conexión con Base de datos

#### Establecimiento de conexión entre base de datos y API

#### Mapeado de Procedimientos

### Conexión Back-End – Front-End

### Implementación de Interfaz Gráfica

### Configuración de la API

# PRUEBAS: UNITARIAS, INTEGRACIÓN Y ACEPTACIÓN

# EJEMPLO DE USO

# CONCLUSIONES

# TRABAJO FUTURO

Dada la magnitud del proyecto, se ha optado por desarrollar un Producto Mínimo Viable (PMV) que incluye las funcionalidades esenciales para asegurar la operatividad solicitada del sistema. Este enfoque permite un lanzamiento rápido y la obtención de retroalimentación temprana de los usuarios.

En las siguientes fases, se implementarán pruebas, incluyendo pruebas unitarias, de integración, de rendimiento y de seguridad, para asegurar la calidad del sistema y evitar operaciones indebidas.

Además, se aplicará un estilo atractivo a la parte de front-end a través de hojas de estilo CSS y la utilización de Boostrap para mejorar la experiencia del usuario.

Asimismo, se integrará la parte actualmente implementada en Access, que maneja las operaciones de SOUCAN, al servicio web.

Esta estrategia garantiza que la aplicación pueda ser ampliada y optimizada de manera continua, incorporando nuevas funcionalidades, mejorando el rendimiento y asegurando una presentación profesional y una gestión de operaciones unificada.

## TRABAJO PREVIO

## FASES DE DESARROLLO