DESARROLLO DE UNA HERRAMIENTA PARA INTERCAMBIO DE INFORMACIÓN MÉDICA BASADA EN ESTÁNDARES

TRABAJO FIN DE GRADO
CURSO 2022-2023

AUTORES
JESÚS ALEJANDRO SÁNCHEZ COUTO
SARA AYUSO DE SANTOS

DIRECTORES
ANTONIO SARASA CABEZUELO
COVADONGA DÍEZ SANMARTÍN

GRADO EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

DEVELOPMENT OF A TOOL FOR EXCHANGING MEDICAL INFORMATION BASED ON STANDARDS

COMPUTER ENGINEERING'S FINAL PROJECT

AUTHORS

SARA AYUSO DE SANTOS

JESÚS ALEJANDRO COUTO SÁNCHEZ

DIRECTORS

ANTONIO SARASA CABEZUELO

COVADONGA DÍEZ SANMARTÍN

CALL: SEPTEMBER 2023

COMPUTER ENGINEERING DEGREE

COMPUTER SCIENCE FACULTY

UNIVERSIDAD COMPLUTENSE DE MADRID

SEPTEMBER 5, 2023

DEDICATORIA

A toda persona que nos haya
acompañado, por que siga haciéndolo

AGRADECIMIENTOS

A los directores de este proyecto, Antonio Sarasa Cabezuelo y Covadonga Díez Sanmartín, por habernos acompañado durante todo el proceso de creación y habernos guiado para que sacáramos adelante este trabajo lo mejor posible.

A nuestros familiares y amigos por habernos acompañado y apoyado en todo momento.

RESUMEN

Desarrollo de una herramienta para intercambio de información médica basada en estándares

Este trabajo consta de la creación de una herramienta para intercambiar información médica basada en estándares entre distintos hospitales, y tiene como objetivo facilitar la compartición de información entre hospitales. La motivación es que cada hospital tiene la libertad de guardar la información relacionada con sus pacientes en bases de datos con formatos completamente diferentes. Esto se produce porque no siguen un estándar específico.

El objetivo de este trabajo consiste en la implementación de una herramienta para facilitar a los hospitales la adaptación de la información clínica de los pacientes procedentes de otros hospitales.

Para conseguir el objetivo anteriormente descrito, la herramienta propuesta permite crear un modelo a partir de la especificación del formato de la base de datos del hospital que la use, además de definir diferentes métricas y unidades de medida y transformar la información de una base de datos externa de acuerdo con el modelo creado.

Palabras clave

Estándar, base de datos, hospital, base de datos de pacientes, aplicación web.

ABSTRACT

Development of a tool for exchanging medical information based on standards

This work consists of the creation of a tool to exchange medical information based on standards between different hospitals, and aims to facilitate the sharing of information between them. This is due to the fact that each hospital has the freedom to keep information pertaining to its patients in completely different databases. This occurs because there is no specific standard.

The objective of this project is to implement a tool able to facilitate the adaptation of other hospital's patient clinical information.

To that end, this tool is able to create a model from a format specification of the hospital's database, as well as to define different metrics and units of measurement and transform an external database's information following the aforementioned model.

Keywords

Standard, database, hospital, patients' database, web application.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capítulo 1 - Introducción

1.1 Motivación

Almacenar y poder acceder a la información de los pacientes es necesario para garantizar un mejor servicio a estos, y las diferencias en cómo se guardan estos datos en diferentes hospitales supone un problema a la hora de compartirlos entre hospitales. En el caso en que algún paciente de un hospital necesitase ser atendido en otro, sería de vital importancia que el hospital donde debe ser tratado acceda a su información médica, y todo ello sin que varíe el formato en el que guarda la información respecto al resto de pacientes.

Para conseguir esto, la solución al problema pasaría por crear una herramienta que sea capaz de obtener el formato en el que guarda la información cada uno de los hospitales entre los que se quiere convertir, así como la equivalencia entre unos campos y otros. Es decir, que si en el modelo de datos de un hospital una cierta información se guarda en un campo con un cierto nombre y en una unidad de medida concreta y el otro hospital lo guarda en un campo que se llama de otra manera y en otra unidad de medida, debería ser capaz de identificarlo y de transformar los datos de un modelo a otro.

La proposición de este proyecto es la de hacer esto de la siguiente manera, que es una aplicación web que permita al hospital usuario crear un modelo de datos propio (además de unidades de medida propias) con los nombres de las columnas. Posteriormente, a la hora de transformar la información, solicita el modelo de datos del hospital externo mediante en archivo CSV, permitiendo seleccionar las distintas filas del modelo del hospital externo y el modelo propio para vincularlas o desvincularlas, además de la unidad de medida de cada fila. Una vez hecho esto, se podrán ordenar las distintas filas del lado izquierdo en una vista previa, confirmar o cancelar e importar los datos para convertirlos.

1.2 Objetivos

El objetivo del proyecto es desarrollar una herramienta que permita a los hospitales intercambiar información de sus bases de datos sin la necesidad de reestructurar su propio sistema.

Esta herramienta debe permitir a los usuarios definir la estructura de su base de datos, así como transformar una base de datos externa de acuerdo a ésta de forma local.

El usuario podrá definir tanto las métricas como las unidades de medida que utilizará, crear el esquema de su base de datos usando dichas métricas y establecer relaciones entre las columnas de un esquema externo y el suyo propio.

1.3 Plan de trabajo

Para la división del trabajo, se ha utilizado una metodología Scrum, autogestionando el trabajo y poniéndose cada parte en común cada 2 semanas.

Para el código se ha utilizado Git Flow, permitiendo trabajar de forma individual en los objetivos de un *sprint* sin poner en riesgo el proyecto, ya que la rama *main* siempre tiene una versión sin errores.

Para la integración continua se ha probado el código de forma manual, puesto que está preparado para funcionar completamente de forma local. Una vez una versión del programa pasa todas las pruebas, se incorpora a la rama *main* de github.

En la *Figura 1* se muestra el Diagrama de Gantt.

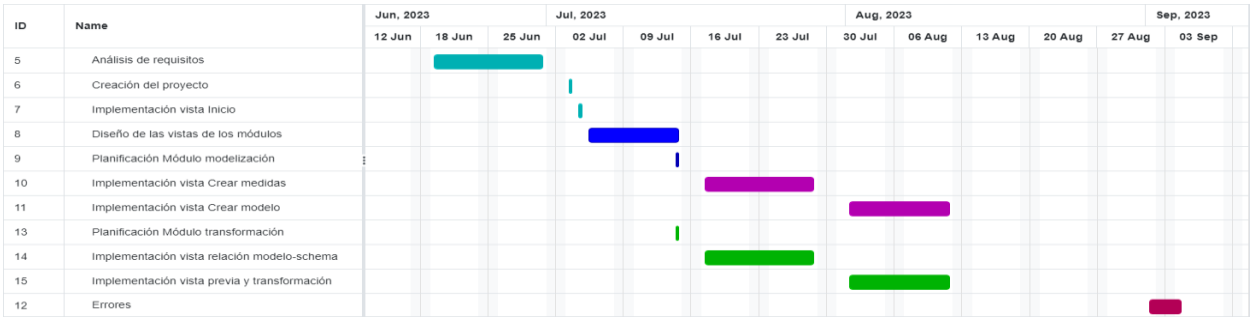

Figura 1- Diagrama de Gantt

Chapter 1 - Introduction

1.1 Motivation

Storing and being able to access patient information is necessary to ensure better service to patients, and differences in how these data are stored in different hospitals is a problem when sharing It between them. In the event that a patient from one hospital needs to be treated in a different hospital, it is vital that the hospital where he is to be treated has access to his medical information, and all this without changing the format in which he keeps information about the rest of patients.

To achieve this, a solution to the problem would be to create a tool that is able to obtain the format in which the information is kept by each of the hospitals between which you want to convert, as well as the equivalence between some fields and others. That is, in the data model of a hospital some information is stored in a field with a certain name and in a specific unit of measure and the other hospital keeps it in a field that is called another way and in another unit of measure, and should be able to know and transform data from one model to another.

The idea of this project is as follows: a web application that allows a hospital to create its own data model (in addition to its own units of measurement) with the names of the columns. Afterwards, when transforming the information, It would ask the data model of another hospital using CSV format. The different rows of the external hospital (to the left) equivalent to one row of the hospital using the application (to the right) are able to be selected and linked or unlinked, in addition to the units of measurement of each row. Once that's done, It would be able to sort the different rows on the left side in a preview and import the database of the external hospital to convert It.

1.2 Goal

This project's goal is to develop a tool that allows hospitals to exchange database information without the need of restructuring its own system.

This tool must be able to let users define the structure of their database in a model, as well as transform an external database locally according to It.

The user will be able to define the quantities as well as their units of measurement, to create the database's scheme using them and to establish relations between the columns of said scheme and an external one.

1.3 Workplan

The methodology used to divide the workload is Scrum, self-managing the work and combining them every 2 weeks.

Git Flow has been used to ensure individual work during a *sprint* doesn't compromise the project, since it ensures the *master* branch always has a working version without errors.

For continuous integration, each individual part of a *sprint* has been tested individually, since It can be fully run locally. A new version is incorporated to the *main* branch once It has passed all the tests.

The *Figure 1* shows the Gantt diagram.

Capítulo 2 - Estado de la cuestión

En este capítulo se describen algunas aplicaciones con funcionalidades que intentan resolver el mismo problema que la aplicación desarrollada en este proyecto.

2.1 Clinical Document Architecture

Clinical Document Architecture o CDA es un estándar de marcado XML desarrollado por HL7 que se centra en la lectura y el envío de documentos clínicos[1]. Sus principales características son:

- Define la estructura de documentos clínicos, especificando cómo se estructura el contenido del documento y cuál puede ser dicho contenido.
- Permite implementar localmente extensiones del estándar para poder compartir información que no está representada en el estándar.
- Los documentos creados pueden ser almacenados independientemente del programa que los utilice y son legibles.

2.2 Digital Imaging and Communications in Medicine

DICOM es un protocolo de estándares para el manejo y el envío de imágenes médicas y su información complementaria[2]. Cuenta con las siguientes características:

- Agrupa la información en *data sets* para poder relacionar las imágenes con el paciente o para enviar información adicional.
- Varias imágenes pueden almacenarse en un solo *data set*.
- Agiliza el flujo de trabajo al automatizar la entrada de los datos del paciente, evitando errores.
- Permite la adición de atributos adicionales que no forman parte del estándar.

2.3 openEHR

Es una organización que define estándares para historias clínicas electrónicas. No tiene una aplicación propia y se centra en la definición de estándares, pero dispone de

varias aplicaciones de código abierto creadas por otros usuarios para implementarlos. Además, el uso de plataformas low-code agilizan el desarrollo de estas aplicaciones y fomenta la interoperabilidad, ya que todas estas aplicaciones utilizan el estándar desarrollado por openEHR[3].

2.4 Fast Healthcare Interoperability Resources

FHIR es un estándar para el envío de información entre sistemas de entidades de la salud desarrollado por *Health Level 7*. Dispone de una RESTful API que permite la lectura, creación, eliminación, modificación y envío de información, y se utiliza principalmente para agilizar la comunicación de aplicaciones con sistemas de HCEs y el cliente. La API utiliza un sistema basado en HTTPS lo que permite herramientas de análisis leer los datos en tiempo real, lo cual puede ser utilizado por entidades de la salud para mantener un seguimiento de epidemias y otras incidencias[4].

Capítulo 3 - Tecnología empleada

Puesto que en el proyecto se han necesitado abarcar varios aspectos del diseño, no solo de la codificación, se ha hecho uso de las tecnologías explicadas en esta sección.

3.1 Visual Studio Code

Visual Studio Code es un IDE, un entorno de desarrollo, que permite crear código de una gran variedad de lenguajes, ejecutar ese código y debuggear, poniendo puntos de interrupción y viendo el estado de todas las variables. Pero, como se explicará en la sección 3.7, se ha usado Github en nuestro proyecto, y Visual Studio Code tiene una sección que permite gestionar los cambios en un repositorio Github, lo cual ha servido de gran ayuda en todo el proceso de desarrollo del código. Además es posible descargarse extensiones que faciliten la tarea y su interfaz es muy intuitiva y fácil de usar. [5]

3.2 HTML5, CSS3, Bootstrap y Javascript

HTML5 es un lenguaje de marcas para crear la estructura de la página web, que junto con CSS3 y el framework de Bootstrap aportando la parte más visual, se ha usado en el proyecto para crear la parte estática de las vistas de la aplicación.

HTML (HyperText Markup Language) es un lenguaje de marcas y etiquetas que permite añadir el contenido de la página entre etiquetas que el navegador pueda interpretar y mostrarlo con el formato deseado. Esto es la estructura, el esqueleto de la página web, de forma que se cambiara la parte visual que la caracteriza, seguirá la misma información. [6]

Esa parte visual se gestiona mediante CSS (Cascading Style Sheets), que como su propio nombre indica, son hojas de estilo. Este lenguaje se centra en aplicar al contenido la forma en que se quiere mostrar, ya sea el color, el tamaño, la distribución, etc. [7]

Bootstrap por su parte, es un framework que aporta un formato visual respecto a muchos componentes HTML. Se incluye mediante un link en el código HTML sobre el que se desea aplicar y se va haciendo referencia a él incluyendo clases específicas en los distintos elementos. [8]

Por último, Javascript es un potente lenguaje de programación que, entre otras cosas, permite añadir de manera dinámica funcionalidades en aplicaciones web. Esto es mediante la creación de scripts con comandos y funciones que se llamen desde las distintas vistas. [9]

Todos estos lenguajes anteriormente descritos se utilizan para desarrollar la parte del cliente de la aplicación.

3.3 Node.js

Es un entorno para crear aplicaciones en Javascript del lado del servidor, y trabaja en tiempo de ejecución, lo cual permite ejecutarlas sin necesidad de hacer uso de un explorador web, directamente en un dispositivo que haga la función de servidor, como por ejemplo un ordenador personal, que puede ser el mismo desde el que se ve la parte del cliente. [10]

Además, mediante su gestor de paquetes Node Package Manager (NPM) permite acceder a paquetes reutilizables ya creados y resolver las dependencias entre ellos. [11]

3.4 Express JS

Express es un framework perteneciente a Node que permite procesar peticiones "middleware" e interpretar peticiones HTTP desde distintas direcciones de vistas como pueden ser por ejemplo GET, POST, etc. También se pueden aplicar ajustes a la aplicación como qué puerto usar o qué motor de renderización de vistas usar, que en nuestro caso hemos optado por EJS. [12]

3.5 AJAX

AJAX (Asynchronous JavaScript And XML) es una tecnología que permite realizar peticiones asíncronas entre el cliente y el servidor. Esto resulta de gran utilidad porque es posible, por ejemplo, obtener en tiempo real datos necesarios en una vista y de esa forma actualizarla mediante Javascript. [13]

3.6 Github

Github es una herramienta para implementar un sistema de control de versiones mediante la creación y actualización de repositorios. Github admite cualquier tipo de archivos pero resulta especialmente útil a la hora de programar, ya que permite administrar las distintas versiones que van surgiendo, creando cuantas ramas se quiera si es necesario y separar así las distintas líneas de desarrollo, con el objetivo de evitar problemas, y luego juntarlo todo en la rama principal, aceptando y rechazando los cambios que se han ido haciendo en cada rama. [14]

3.7 IBM RSAD

IBM Rational Software Architect Designer es un entorno dedicado al diseño de un software, que permite elaborar los distintos diagramas que puedan intervenir, como por ejemplo los diagramas de casos de uso, diagramas de secuencia o diagramas de clases entre muchos otros. Son diagramas en lenguaje UML, el cual es universal. [15]

Capítulo 4 - Especificación de requisitos

4.1 Actores

Al estar enfocada la aplicación en adaptar cualquier base de datos y en cualquier hospital, está hecha para que la use aquella o aquellas personas que se encargan de la gestión de los datos de los pacientes. Por ello no hay un inicio de sesión, así que no hay distintos tipos de usuarios. Es decir,la aplicación tiene un único actor, que en este caso es el ADMINISTRADOR.

4.2 Módulos

De acuerdo a las funcionalidades de la aplicación, se distinguen los siguientes módulos:

- Módulo de modelización
- Módulo de transformación

4.2.1 Módulo de modelización

En este módulo se incluyen todos aquellos casos de uso que hacen referencia a la creación del modelo de la base de datos del hospital que usa la aplicación.

Diagrama de casos de uso:

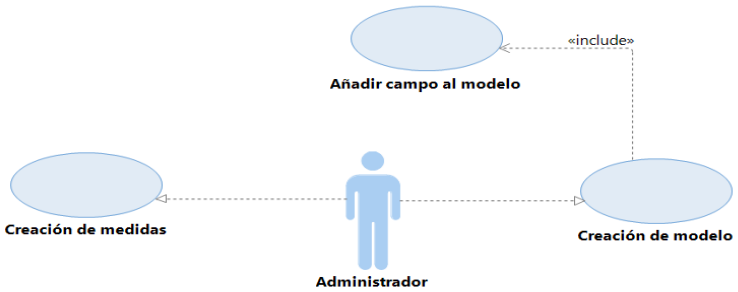

Figura 2- Diagrama de casos de uso Módulo modelización

A continuación se analiza cada caso de uso:

- Diagrama de Creación de medidas

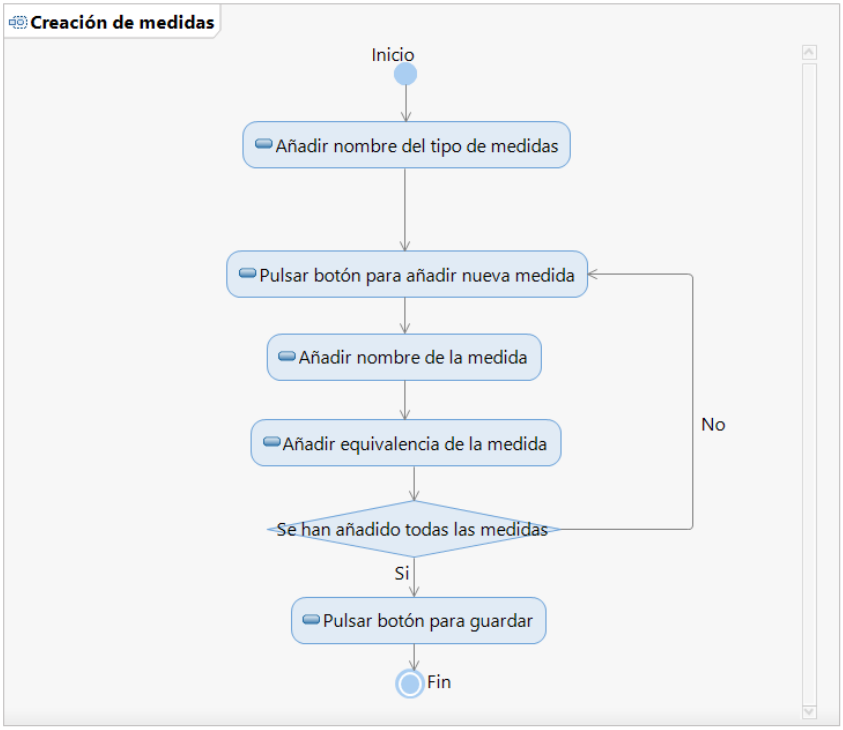

Figura 3- Diagrama de actividad Creación de medidas

FUNCIÓN	Creación de medidas
DESCRITA POR	Sara Ayuso de Santos
PRIORIDAD	Media
ESTABILIDAD	Alta
DESCRIPCIÓN	Se crea un bloque de medidas del mismo tipo para usar al crear el modelo.
ACTOR	Administrador
ENTRADA	No hay

SALIDA	Texto plano con las medidas creadas.
ORIGEN	Interfaz de usuario
DESTINO	Archivo medidas.txt
NECESITA	Nada
ACCIONES	1. Se introduce el nombre del tipo de las medidas que vamos a crear. 2. Se introduce una nueva medida, con su nombre y su equivalencia. 3. Si se quisiera seguir añadiendo, se repite el paso 2 hasta que no queramos añadir más medidas. 4. Si se ha terminado, se da al botón de guardar. 5. Si no existe el archivo medidas.txt, se crea. 6. Se añade el texto de las nuevas medidas al final del archivo medidas.txt.
PRECONDICIÓN	No hay
POSCONDICIÓN	En el caso de que no existiera el archivo medidas.txt, se habrá creado. Y en cualquier caso, se habrá añadido el texto a ese archivo.
EFECTOS LATERALES	No hay

Tabla 1- Creación de medidas

- Diagrama de Creación de modelo

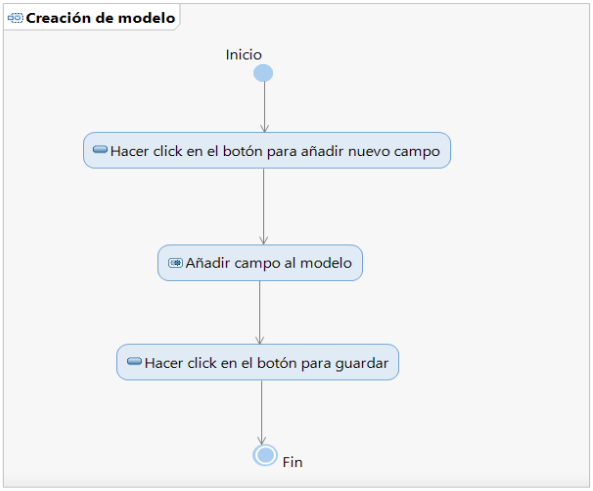

Figura 4- Diagrama de actividad Creación de modelo

FUNCIÓN	Creación de modelo
DESCRITA POR	Sara Ayuso de Santos
PRIORIDAD	Alta
ESTABILIDAD	Alta
DESCRIPCIÓN	Se crea un archivo con los campos del modelo de la base de datos del hospital.
ACTOR	Administrador
ENTRADA	Cada uno de los campos que se desean añadir.
SALIDA	Archivo en formato CSV que describe el formato del modelo creado.
ORIGEN	Interfaz de usuario
DESTINO	Archivo modelo.csv
NECESITA	Nada
ACCIONES	1. Se añade un campo nuevo. 2. Si se quieren añadir más campos se repite el paso número 1 hasta que no se quieran añadir más. 3. Si ya se ha terminado se da al botón de guardar. 4. Si no existe el archivo modelo.csv se crea uno nuevo. 5. Se sobrescribe el texto que haya en el archivo modelo.csv, puesto que un hospital solo tiene un modelo de base de datos.
PRECONDICIÓN	No hay
POSCONDICIÓN	Se habrá creado o sobreescrito el archivo modelo.csv.
EFECTOS LATERALES	No hay

Tabla 2- Creación de modelo

- Diagrama de Añadir campo al modelo

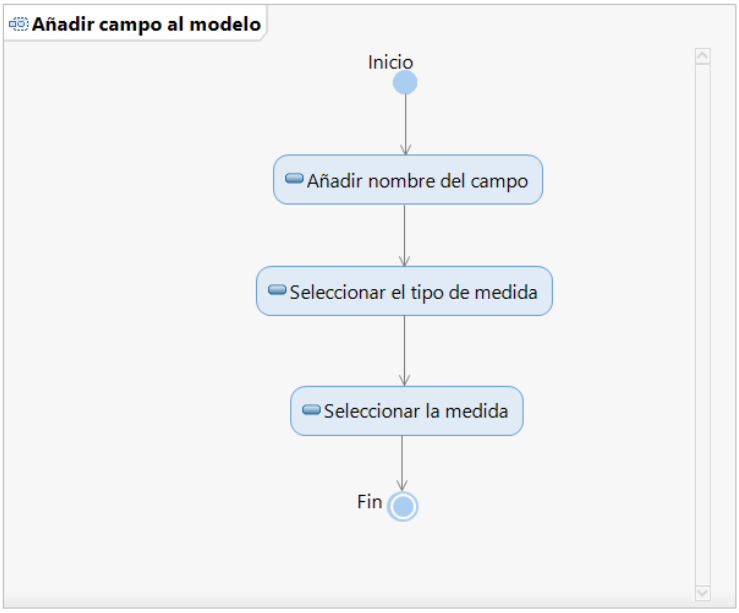

Figura 5- Diagrama de actividad Añadir campo al modelo

FUNCIÓN	Añadir campo al modelo
DESCRITA POR	Sara Ayuso de Santos
PRIORIDAD	Media
ESTABILIDAD	Alta
DESCRIPCIÓN	Se añade un campo nuevo al modelo de la base de datos.
ACTOR	Administrador
ENTRADA	Nombre del campo, tipo de la medida y nombre de la medida del campo.
SALIDA	No hay
ORIGEN	Interfaz de usuario
DESTINO	Interfaz de usuario
NECESITA	Nada
ACCIONES	1. Se añade el nombre. 2. Se selecciona el tipo de medida. 3. Según el tipo seleccionado se muestran unas medidas u otras en el segundo selector.

	4. Se selecciona la medida del campo.
PRECONDICIÓN	Deben existir las medidas y los tipos de medidas que se quieran añadir a los campos, y en caso de que no existieran se deben haber creado.
POSCONDICIÓN	Se habrá añadido un campo nuevo al modelo.
EFECTOS LATERALES	En el caso de que el tipo o la medida que se quiera añadir no se haya creado no va a dar opción a seleccionarla.

Tabla 3- Añadir campo al modelo

4.2.2 Módulo de transformación

Este módulo abarca aquellos casos de uso que se refieren a la importación de los datos de un hospital externo y la adaptación de los mismos a la base de datos propia del hospital.

Diagrama de casos de uso:

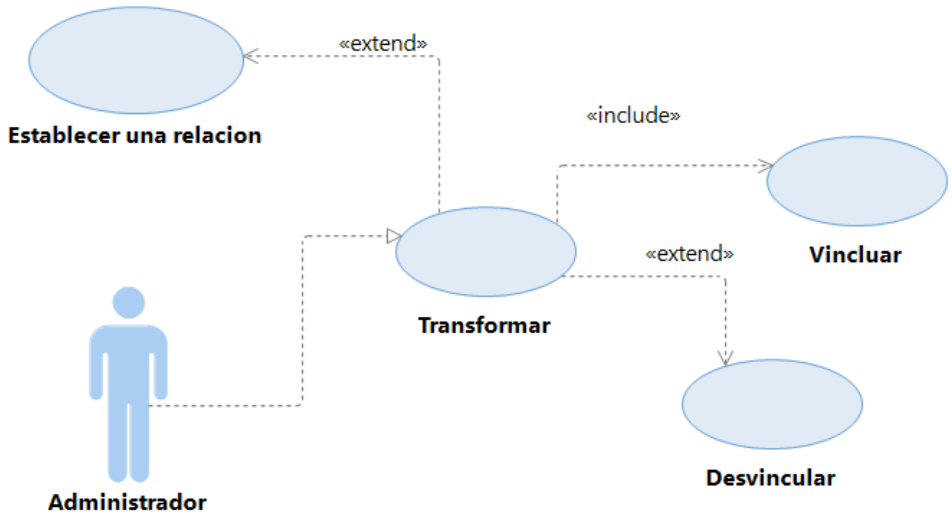

Figura 6- Diagrama de Casos de uso Módulo transformación

Ahora se analiza cada caso de uso:

-Diagrama de actividad de Establecer una relación

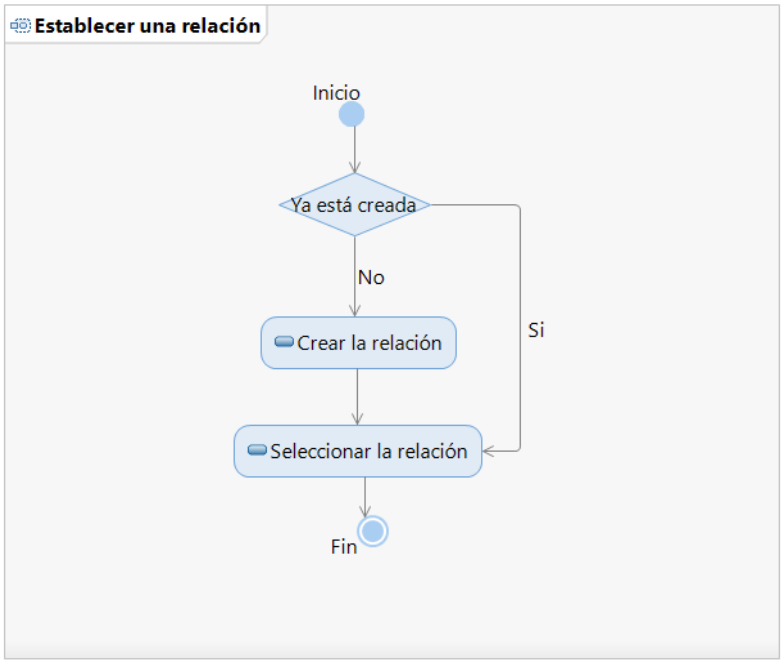

Figura 7- Diagrama de actividad Establecer una relación

FUNCIÓN	Establecer una relación
DESCRITA POR	Jesús Alejandro Sánchez
PRIORIDAD	Alta
ESTABILIDAD	Media
DESCRIPCIÓN	Relaciona filas del modelo externo con el esquema de acuerdo a una opción
ACTOR	Administrador
ENTRADA	Opción del selector de relaciones
SALIDA	Visualización de la relación entre las filas del modelo externo y el esquema
ORIGEN	Interfaz del usuario

DESTINO	Interfaz del usuario y datos de relaciones
NECESITA	Haber importado el modelo externo
ACCIONES	1. El administrador abre el selector y escoge una de las opciones. Si es una relación ya existente, se cargan los datos y se representan en pantalla 2. Se realiza una de las siguientes: 2.1. Se escoge la opción "Create New" y aparece un cuadro donde se introduce el nombre de la nueva relación 2.1.1. El administrador hace clic en confirmar y se crea la nueva relación 2.1.2. El administrador hace clic en cancelar y se cierra el cuadro, finalizando la operación 2.2. El administrador escoge una de las relaciones existentes 3. Se carga la relación seleccionada y se representa en pantalla
PRECONDICIÓN	Se ha importado un modelo de una base de datos externa
POSCONDICIÓN	Se representan visualmente las filas que tienen relaciones del modelo externo y el esquema de acuerdo con la información de la relación seleccionada
EFECTOS LATERALES	No hay

Tabla 4- Establecer una relación

-Diagrama de actividad de Vincular

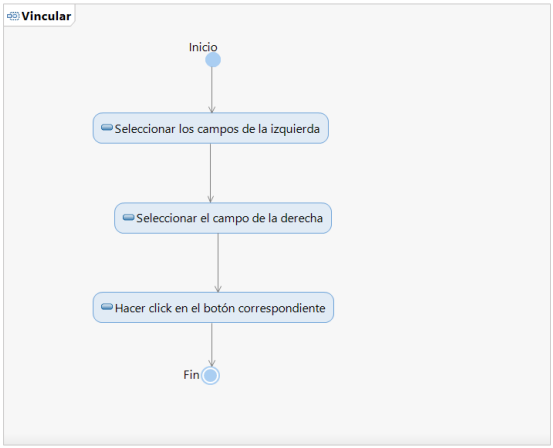

Figura 8- Diagrama de actividad Vincular

FUNCIÓN	Vincular
DESCRITA POR	Jesús Alejandro Sánchez
PRIORIDAD	Alta
ESTABILIDAD	Media
DESCRIPCIÓN	Relaciona filas del modelo externo con una fila del esquema
ACTOR	Administrador
ENTRADA	Filas seleccionadas
SALIDA	Actualización y visualización de las relaciones entre las filas seleccionadas
ORIGEN	Interfaz del usuario
DESTINO	Interfaz del usuario
NECESITA	Haber importado el modelo externo
ACCIONES	1. El administrador hace clic en los elementos de la fila y el elemento del esquema que quiere relacionar 2. El administrador hace clic en el botón "Vincular" 3. Se vinculan las filas seleccionadas y se representa visualmente en pantalla
PRECONDICIÓN	Las filas seleccionadas no están en una relación y las métricas de las filas que no sean "Texto" coinciden.
POSCONDICIÓN	Se representa visualmente la relación entre las filas seleccionadas
EFECTOS LATERALES	Si se pulsa en vincular no se cumple la precondición, no se realiza ninguna acción.

Tabla 5- Vincular

-Diagrama de actividad de Desvincular

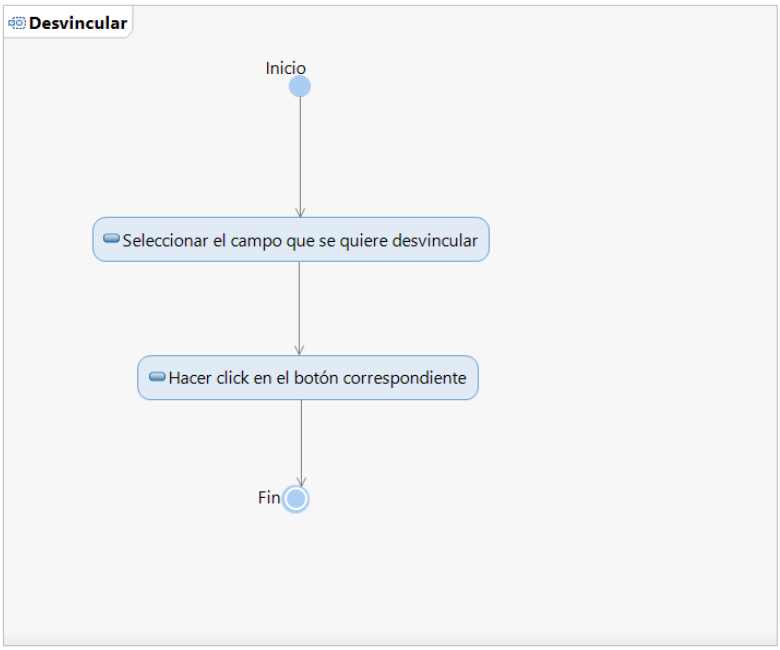

Figura 9- Diagrama de actividad Desvincular

FUNCIÓN	Desvincular
DESCRITA POR	Jesús Alejandro Sánchez
PRIORIDAD	Alta
ESTABILIDAD	Media
DESCRIPCIÓN	Desvincula filas del modelo externo con una fila del esquema
ACTOR	Administrador
ENTRADA	Filas seleccionadas
SALIDA	Actualización y visualización de las relaciones entre las filas relevantes
ORIGEN	Interfaz del usuario
DESTINO	Interfaz del usuario
NECESITA	Haber importado el modelo externo
ACCIONES	1. El usuario hace clic en una fila

	2. El usuario hace clic en el botón "Desvincular" 3. Se desvinculan las filas pertinentes y se representa visualmente
PRECONDICIÓN	La fila seleccionada debe estar en una relación
POSCONDICIÓN	Se representa visualmente la relación entre las filas seleccionadas
EFECTOS LATERALES	No hay

Tabla 6- Desvincular

-Diagrama de actividad de Transformar

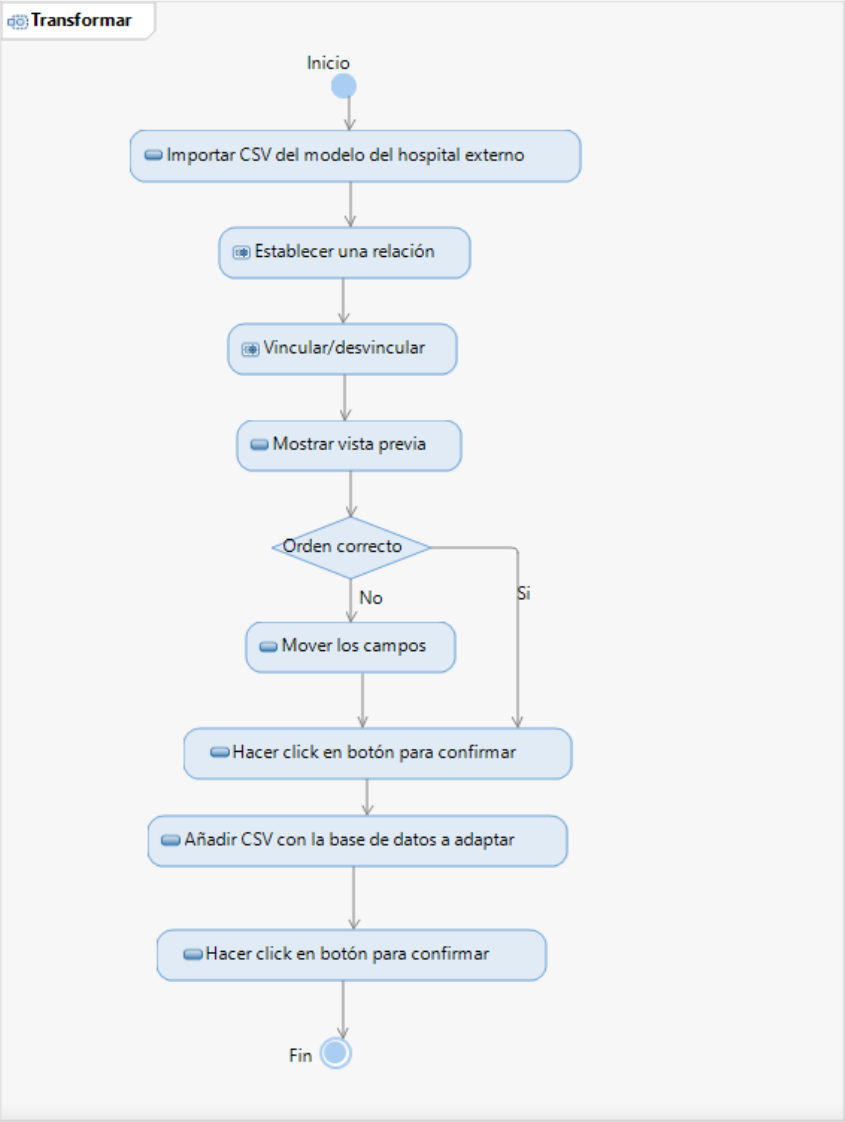

Figura 10- Diagrama de actividad Transformar

FUNCIÓN	Transformar
DESCRITA POR	Jesús Alejandro Sánchez
PRIORIDAD	Alta
ESTABILIDAD	Media
DESCRIPCIÓN	Adapta una base de datos externa de acuerdo a las relaciones de su modelo con el esquema

ACTOR	Administrador
ENTRADA	Base de Datos Externa
SALIDA	Descarga de la nueva Base de Datos con los datos transformados de acuerdo a las relaciones entre filas
ORIGEN	Interfaz del usuario
DESTINO	Interfaz del usuario
NECESITA	Que haya al menos una relación entre filas del modelo externo y el esquema
ACCIONES	1. El administrador hace clic en el botón "Transformar" 2. Aparece una vista previa clarificando cómo se van a transformar las filas y a qué fila del esquema van 3. El administrador hace clic sobre las filas de tipo "Texto" para ordenarlas 4. Se procede por una de las siguientes: 4.1. El administrador hace clic en "Confirmar" 4.1.1. En el cuadro de la preview aparece un Drag and Drop 4.1.2. El administrador desplaza el archivo de la base de datos sobre el Drag and Drop 4.1.3. Se descarga la base de datos transformada 4.2. El administrador hace clic en "Cancelar" 5. Se cierra la ventana de preview
PRECONDICIÓN	Que haya al menos una relación entre las filas del modelo externo y el esquema
POSCONDICIÓN	Se representa visualmente la relación entre las filas seleccionadas
EFECTOS LATERALES	Si el archivo que se desplaza sobre el Drag and Drop no es un csv, no ocurre nada.

Tabla 7- Transformar

Capítulo 5 - Arquitectura y modelo de datos

5.1 Arquitectura

La arquitectura de la aplicación encajaría con un patrón cliente-servidor, pero el servidor es local y hay un solo cliente (*Figura 11*). Este patrón se caracteriza por tener dos componentes: el cliente, que realiza peticiones, y el servidor, que escucha las peticiones del cliente y le proporciona ciertos servicios. [16]

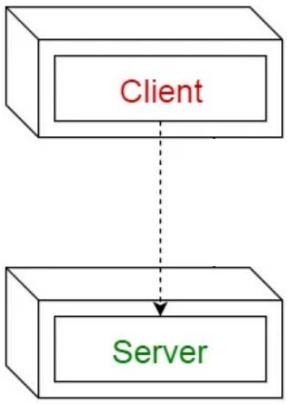

Figura 11- Modelo cliente-servidor

5.2 Modelo de datos

En lugar de una única base de datos con toda la información, los datos se almacenan en diferentes archivos que hacen la función de bases de datos dependiendo de su finalidad. Es decir, serían bases de datos orientadas a documentos.

La información a almacenar consiste de: las métricas junto con sus diferentes unidades de medida, que se usarán para indicar las unidades de medida de cada columna tanto del esquema como del esquema cuya base de datos se quiere adaptar, y la relación entre las diferentes columnas de la base de datos externa y el esquema de la aplicación, habiendo un fichero por cada relación.

Las unidades de medida se guardarían en la entidad MEDIDA del diagrama E-R (*Figura 12*), y el tipo guardado como atributo en las medidas serían las métricas. Además

se guarda cada campo del modelo creado en la entidad CAMPO, el cual tiene asociado un nombre y una medida, que por eso guardan relación en el diagrama.

De acuerdo a lo anterior, el modelo se guarda en un documento, el cual está en formato CSV y las columnas guardan el orden de "Nombre", "Tipo" y "Medida".

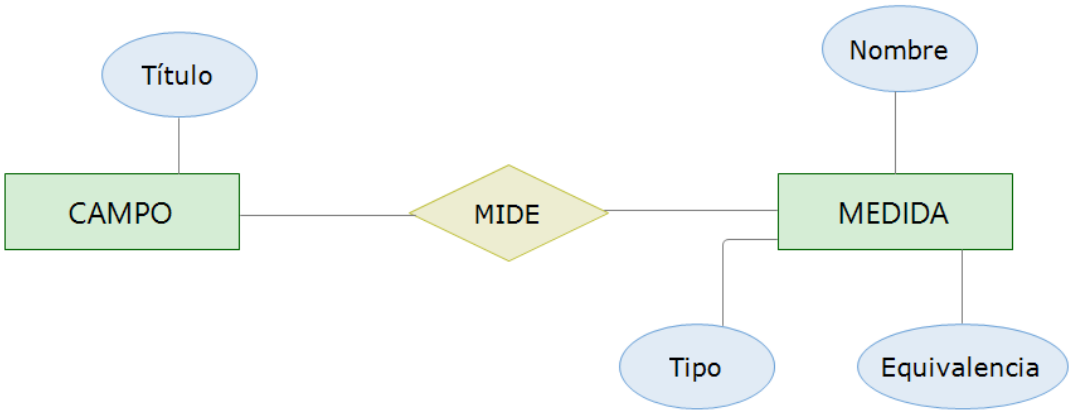

Figura 12- Diagrama Entidad-relación

5.2.1 Modelo

Esta colección se utiliza para almacenar el esquema con las columnas de la base de datos final. En ella se almacena, por cada elemento creado, un nombre, la métrica que le corresponde y su unidad de medida (*Figura 13*). Se utiliza para poder cargar el esquema a la hora de querer transformar una base de datos externa.

```
0: "Columna0;Tiempo;minutos"
1: "Columna1;Peso;gramos"
2: "Columna2;Temperatura;Kelvin"
3: "Columna3;Texto;texto"
```

Figura 13- Captura modelo creado

5.2.2 Medidas

Esta colección se utiliza para almacenar las métricas creadas y sus unidades de medida. En ella se almacena, por cada elemento creado, el nombre de la métrica, el

nombre de las unidades de medida que les corresponden y una operación aritmética por cada unidad de medida, que representa su relación con la primera unidad de medida de la métrica (*Figura 14*). Se utiliza para poder indicar tanto las medidas del esquema al crearlo como las de la base de datos a transformar.

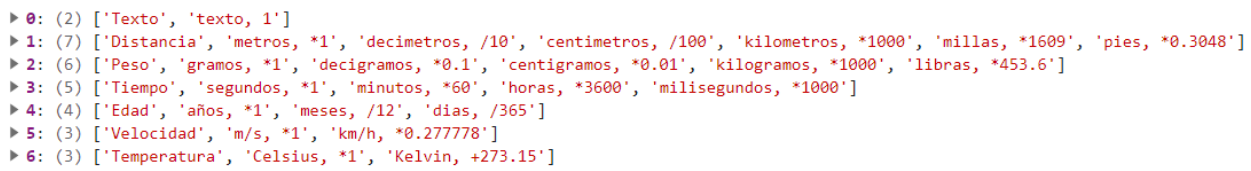

Figura 14- Captura medidas creadas

5.2.3 Conjunto de Relaciones

Hay tantos ficheros con este tipo de colección como conjuntos de relaciones para una transformación se hayan creado (*Figura 15*). Esta colección contiene el estado de las relaciones de todas las columnas de la base de datos a transformar y el esquema, así como las relaciones en sus elementos. En ella se almacena:

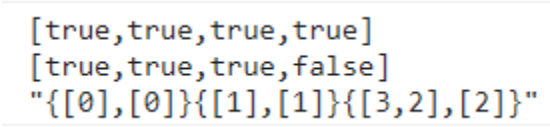

Figura 15- Captura relaciones creadas

-Por cada elemento de la base de datos a transformar, si está vinculado con algún elemento del esquema.

-Por cada elemento del esquema, si está vinculado con algún elemento de la base de datos a transformar.

-Todas las relaciones, indicando qué elementos de la base de datos están relacionados con qué elementos del esquema.

Esta colección se utiliza para poder cargar y representar visualmente las relaciones entre los elementos de ambas partes al cambiar el valor selector de la vista.

Capítulo 6 - Diseño e implementación

En este capítulo también vamos a hacer una diferenciación entre los módulos de acuerdo a las distintas funcionalidades que ofrece cada uno de ellos:

- Módulo de modelización
- Módulo de transformación

Y ahora describiremos el diseño y la implementación de cada uno en detalle.

6.1 Módulo de modelización

Incluye todas las funcionalidades que se utilizan para preparar el esquema, a las que se puede acceder desde "Crear medidas propias" y "Crear un modelo propio".

6.1.1 Creación de una métrica

La vista que se visualiza al acceder a la ventana "Crear medidas propias". Contiene un campo "Tipo", que es único, un botón para añadir unidades de medida y un botón para guardar la métrica. Cada vez que se hace clic en "Añadir medida", se crea un elemento debajo del campo "Tipo" con dos nuevos campos vacíos, "Medida" y "Equivalencia". Después de rellenar los campos, hacer clic en "Guardar" guarda en el archivo medidas el valor del campo "Tipo" seguido de todos los campos de acuerdo con su formato, tal y como se indica en la *Figura 16*.

```
function submitFormulario(event) {

    event.preventDefault();

    const tipo = document.querySelector('[name="tipo"]').value;
    const medidas = document.querySelectorAll('[name="medidas[]"]');
    const equivalencias = document.querySelectorAll('[name="equivalencias[]"]');
    var text = `${tipo}\n`;

    for(let i=0; i<medidas.length; i++) {
        text += medidas[i].value + ', ';
        text += equivalencias[i].value + '\n';
    }
     text += '--\n';
    fetch('/crear_medidasS', {
      method: 'POST',
      headers:{
          'Content-Type': 'application/json'
      },
      body: JSON.stringify({text: text})
  })
  .then(response => response.text())
  .then(message => {
      console.log(message);
  })
  .catch(error => {
      console.error("There's been an issue creating the links file: "+ error);
  });
```

Figura 16- Código guardar una métrica

6.1.2 Creación del esquema

Al hacer clic en "Crear un modelo propio", la aplicación carga una ventana con los campos correspondientes a un elemento del esquema y los botones para añadir nuevos elementos y guardar el esquema. Las métricas y sus unidades de medida se leen del fichero creado en el punto anterior en el momento que la página termina de cargar, añadiendo los nombres de cada métrica al primer selector. De acuerdo a la *Figura 17*, cambiar la opción del primer selector de un elemento llama a una función que vacía el segundo selector y añade las nuevas opciones con las unidades de medida correspondientes. Finalmente, una vez rellenados los campos y sus selectores, hacer clic en guardar almacena los datos en un archivo csv delimitado por puntos y comas, indicando en cada fila el nombre del elemento, su métrica de acuerdo con el primer selector y la unidad de medida del segundo.

```
function llenarNombres(element) {
    var pos = quantities.indexOf(element.target.value);
    console.log(pos);
    console.log(element);
    var m = element.target.nextSibling.nextSibling.nextSibling.nextSibling;
    m.innerHTML = '';
    var metrics = options[pos];
    for(var i = 1; i<metrics.length;i++){
        var option = document.createElement("option");
        option.value = metrics[i].split(',')[0];
        option.text = metrics[i].split(',')[0];
        m.appendChild(option);
    }
}
```

Figura 17- Código crear nuevas opciones

6.2 Módulo de transformación

Al hacer clic en el botón "Adaptar un modelo externo", la aplicación carga una vista con un selector vacío y un *input* para cargar el modelo externo, que debe tener formato csv y estar delimitado por comas. Seleccionar el modelo externo cargará todas sus filas en la mitad izquierda con los selectores de las métricas y sus unidades de medida, todos los elementos del esquema en la mitad derecha y la lista de conjuntos

de relaciones en el selector vacío de acuerdo con la *Figura 18*, en ese orden.

```
function loadSelect(){
    var a = document.getElementById('maps');
    for(var i = a.options.length; i> 0;i--){
        a.remove(i);
    }
    a.options[0].selected = true;
    fetch('/relationsS')
        .then(response => response.json())
        .then(data => {
            data.forEach(element => {
                var a = document.getElementById('maps');
                var option = document.createElement('option');
                var ele = String(element)
                ele = ele.replace(".txt", "");
                option.textContent = ele;
                option.value = ele;
                a.appendChild(option);
            });
            var a = document.getElementById('maps');
            var option = document.createElement('option');
            option.textContent = "Create New";
            option.value = "create";
            a.appendChild(option);
            var event = new Event("select_loaded");
            document.dispatchEvent(event);
        })
        .catch(error => {
            console.error("Couldn't read the options")
        });
}
```

Figura 18- Código cargar relaciones

6.2.1 Relacionar filas

Una vez se ha escogido o creado una opción en el selector, cada fila de ambos modelos se puede seleccionar haciendo clic en ellas. Se pueden seleccionar varias filas de la izquierda y solo una de la derecha, no ocurriendo nada si se intentan seleccionar más. El botón para relacionarlas se habilita si hay al menos una fila de cada lado seleccionada, y al pulsarlo se realiza una serie de comprobaciones, comprobando que ninguna de las filas seleccionadas estuviera ya en una relación existente, que ninguno de los modelos no tuviera filas seleccionadas y que las métricas que no sean texto coincidan, como se aprecia en la *Figura 19*. Para desvincular filas, seleccionar una fila que ya está relacionada con otras las resalta y habilita el botón de desvincular, que deshace la relación.

```
function link(){
    var error = "";
    if(!alreadyLinked(error) && !mismatched(error) && selecteds_length(error)){
        var tr = document.getElementById('transform');
        if(tr.disabled == true) tr.disabled = false;
        var left;
        var right;
        var tmp_L = selectedL.slice();
        var tmp_R = selectedR.slice();
        var both = {left: tmp_L, right: tmp_R};
        links.push(both);
        i = 0;
        while(i<selectedL.length || i<selectedR.length){
            if(i<selectedL.length) {
                left += selectedL[i].toString() + " ";
                linkedL[selectedL[i]] = true;
                var valL = "leftdiv"+selectedL[i];
                var contentL = document.getElementById(valL);
                contentL.classList.remove("selected");
                contentL.classList.add("linked");
            }
            if(i<selectedR.length) {
                right += selectedR[i].toString() + " ";
                linkedR[selectedR[i]] = true
                var valR = "rightdiv"+selectedR[i];
                var contentR = document.getElementById(valR);
                contentR.classList.remove("selected");
                contentR.classList.add("linked");
            }
            i++;
        }
        selectedR.length = 0;
        selectedL.length = 0;
        numL = 0;
        numR = 0;
        document.getElementById("tolink").disabled = true;
    }
    else {console.log(error);}
}
```

Figura 19- Código vincular filas

6.2.2 Transformar

Una vez haya al menos una relación entre filas, se habilita el botón de transformar. Pulsarlo guarda el conjunto de relaciones en un archivo de la forma que indica la figura [], lee todas las relaciones y las representa en filas dentro de un nuevo modal, separando de izquierda a derecha según sea texto del modelo externo, otra métrica del modelo externo o un elemento del esquema, en ese orden. Los elementos texto del modelo externo son seleccionables, pudiendo intercambiar la posición de dos elementos de la misma fila como se indica en la *Figura 20*.

```
function toggleSelect(element){
    //checks if the element was already selected, or, if not, if it can be selected
    //if it was selected
    if(chosen && element.target.classList.contains('pselected')){
        element.target.classList.remove('pselected');
        chosen = false;
    //it wasn't seleced but it can be selected
    } else if(!chosen && !element.target.classList.contains('pselected')){
        element.target.classList.add('pselected');
        chosen = true;
    //if there was already a chosen element and it's from the same row, however, they'll swap places
    } else if(chosen && !element.target.classList.contains('pselected')){
        var a = document.getElementsByClassName('pselected')[0];
        var b = a.id.replace('prev', '')[0];
        var c = element.target.id.replace('prev', '')[0];
        //if the elements share the same parent, they swap places
        //this means they are both in the same row and are both either text or a quantity
        if(a.parentElement == element.target.parentElement){
            swapPlaces(element);
        }
        //regardless of the outcome, the chosen element is unselected
        a.classList.remove('pselected');
        chosen = false;
    }
}
```

Figura 20- Código intercambiar posiciones

Pulsar el botón cancelar elimina el modal, mientras que el botón confirmar cambia su contenido por un *Drag and Drop*. Este elemento comprueba el tipo del archivo que se desplaza sobre él y, si su formato es csv, procede con la transformación. La transformación crea un nuevo archivo csv con las columnas del esquema y los valores de cada columna de cada fila de acuerdo a las relaciones establecidas como se muestra en la *Figura 21*.

```
//saves the arrays links, linkedL and linkedR in a file to save the relations between rows
//then it creates a new file with the transformed csv for the user to download
function transform(){
    var option = document.getElementById('maps').value;
    if(option != "empty" && option != "create"){
        if(links.length > 0){
            //const fix = links.map(item => JSON.stringify(item)); //fix so the server can handle the array of objects properly
            var add = "";
            for(var i = 0;i<links.length;i++){
                add += JSON.stringify(links[i]);
            }
            const a = /"left":|\s*"right":/g;
            const fix = add.replace(a, "");
            const arrays = {
                linkL: linkedL,
                linkR: linkedR,
                link: fix
            };

            var name = document.getElementById("maps");
            const path = "./relations/"+name.value;
            const send = JSON.stringify({fileName: path, data: arrays});
            fetch('relations', {
                method: 'POST',
                headers:{
                    'Content-Type': 'application/json'
                },
                body: send
            })
            .then(response => response.text())
            .then(message => {
                console.log(message);
            })
            .catch(error => {
                console.error("There's been an issue creating the links file: "+ error);
            });
        }
```

Figura 21- Guardar las relaciones

Capítulo 7 - Conclusiones y trabajo futuro

La aplicación se centra en la creación de un modelo que represente la información guardada en la base de datos de un hospital a partir del almacenamiento de los nombres de cada columna y de la medida que se usa para guardar los datos, así como de la transformación de modelos de otros hospitales con el fin de obtener información de pacientes de esos hospitales en el hospital que esté usando esta herramienta.

El resultado es una aplicación sencilla pero muy útil en el campo de la medicina; aunque es una herramienta a la cual no pueden acceder directamente los médicos o los pacientes. En principio está dirigida a un usuario muy concreto y no a ellos, pero sería interesante poder añadir en un futuro alguna funcionalidad para que ellos también puedan utilizarla.

Una posible mejora que añadir en un futuro sería que los médicos vean el historial de todos sus pacientes o que los pacientes vean su propia información o citas médicas que tengan pendientes. Pero también se podría adaptar a muchos otros campos en los que se necesiten adaptar bases de datos, como por ejemplo en un colegio con las notas de los alumnos de las distintas asignaturas, o simplemente en cualquier negocio de cambio de divisas, que necesite saber la equivalencia entre los valores de las divisas.

Todo el código está disponible en GitHub:

(https://github.com/Jesusasa/Estandarizador)

Chapter 7 - Conclusions and future work

The application focuses on creating a model that represents the information stored in a hospital database from the storage of the names of each column and the measure used to save the data, as well as the transformation of models of other hospitals in order to obtain information from patients of those hospitals in the hospital that is using this tool.

The result is a simple but very useful application in the field of medicine; although it is a tool that doctors or patients cannot directly access. In principle it is aimed at a very specific user and not at them, but it would be interesting to be able to add some functionality in the future so that they can also use it.

One possible improvement to add in the future would be for doctors to look at the history of all their patients or for patients to see their own information or medical appointments they have pending. But it could also be adapted to many other fields in which databases need to be adapted, such as in a school with the marks of the students of the different subjects, or simply in any foreign exchange business, you need to know the equivalence between currency values.

All code is available on GitHub:

(https://github.com/Jesusasa/Estandarizador)

Capítulo 8 - Contribuciones personales

Sara Ayuso de Santos

Al haber dos módulos principales en nuestro proyecto, yo me he encargado principalmente de uno de ellos, el de modelización. Pero también de modificar ciertas partes gráficas de la interfaz.

A continuación enumeraré cada una de mis aportaciones en este proyecto más en concreto:

- Creación del proyecto con Node.js mediante npm init, del título y de los módulos necesarios de incluir para el correcto funcionamiento del proyecto.
- Creación del archivo que gestiona mediante Javascript la parte del servidor, llamado main.js, en el que usamos Express mediante paquetes incluidos gracias a Node.
- Configuración del archivo main.js para incluir el uso de todos los paquetes necesarios, como por ejemplo Express, body-parser o csv-paser entre otros.
- Creación y posterior inclusión en main.js del archivo de configuración del puerto y la conexión entre el cliente y el servidor, así como el manejador app.listen.
- Inclusión en el archivo main.js de los manejadores de todas las peticiones por parte del cliente desde las distintas rutas de las vistas que yo he creado mediante GET y POST entre otras cosas.
- Creación de la ventana principal de inicio, con los enlaces a cada una de las tres vistas mediante botones distribuidos con cierto espacio entre medias, usando el framework de Bootstrap.
- Implementación de la ventana de crear nuevas métricas. Contiene un formulario con un input inicial y un botón para añadir nuevas medidas, que en principio no aparece ninguna, dado que se crean de forma dinámica.
- Creación del script para añadir nueva métrica. Este incluye la función para crear dinámicamente los campos a rellenar para las nuevas medidas junto con sus

operaciones aritmético lógicas, así como la función que, cuando se pulsa el botón de guardar, crea un archivo con la información de la métrica creada o escribe esa información al final en caso de que ya exista, es decir, que haya otras medidas creadas.

- Implementación de la vista para crear un nuevo modelo, en la que se pide el nombre, el tipo y la medida de cada campo que se quiera incluir en el modelo.
- Creación del script que se encarga de rellenar los nombres de los selects de la vista de crear modelo. Es decir, en esta vista, según el tipo que selecciones (métrica), en el segundo *select* se mostrarán unas medidas u otras. Solo se muestran las medidas del tipo seleccionado.
- Personalización de los títulos de las vistas, centrándolos y escogiendo la fuente y tamaño.
- Inclusión del botón con la flecha hacia la izquierda en el título de cada vista, que lleva a la vista inicial.

Jesús Alejandro Sánchez Couto

Mi principal contribución en este proyecto ha sido la creación de la ventana "Adaptar un modelo externo", realizar pruebas sobre cada una de las funcionalidades y solventar los errores que surjan entre esta y el resto de la aplicación en cada versión.

- La creación de la base visual de la ventana. Esto incluye todo lo que se encuentra por debajo de la cabecera: el selector de relaciones, el input del modelo externo, los cuadros donde se cargan tanto el modelo con sus selectores como el esquema y los botones flotantes para vincular, desvincular y transformar.
- Funcionalidad de la carga del modelo externo y el esquema en la página. Esto incluye tanto la lectura de ambos ficheros como la creación de las filas de cada uno según los elementos leídos.
- Funcionalidad para cargar la lista de conjuntos de relaciones una vez se han cargado el modelo y el esquema. Esto incluye tanto la lectura de los nombres de los archivos como la creación de las opciones en el selector según los archivos leídos.

- Funcionalidad de la carga automática de las métricas y de las unidades de medida en los selectores correspondientes de cada fila, así como la actualización de las unidades de medida cuando se cambia el valor del selector de métrica. Esto incluye tanto la lectura de las unidades de medida correspondientes en función de la métrica seleccionada como la creación de las nuevas opciones en el selector derecho.
- Funcionalidad para vincular y desvincular conjuntos de filas y la lógica que previene. Esto incluye la lógica que permite seleccionar y deseleccionar filas, así como la representación visual en función del estado de la fila en el momento que se hace clic en ellas, y la actualización del estado de las mismas cuando se pulsan los botones "Vincular" y "Desvincular" según convenga.
- Funcionalidad para vincular las filas correspondientes cuando se selecciona un conjunto de relaciones en el selector. Esto incluye la lectura del archivo correspondiente, la actualización del estado de las filas de acuerdo con la información del fichero y su representación visual correspondiente.
- Funcionalidad para habilitar los diferentes botones cuando se cumplen las condiciones para utilizarlos y prevenir que no se puedan usar en otros casos.
- La creación de la ventana de la vista previa, incluyendo la lógica de los botones, la representación visual de las filas del modelo externo y el esquema y la funcionalidad de reordenarlas haciendo clic en ellas.
- La creación del cuadro de *drag and drop*, así como la lógica que permite identificar el fichero que se desplaza sobre él y proceder con la transformación de éste en caso de cumplir las condiciones. Esto incluye la lectura de los elementos del fichero, la transformación de sus valores de acuerdo a las relaciones creadas y las métricas de las filas del modelo y el esquema, así como la creación del fichero final y su descarga.
- La inclusión de la lógica pertinente en el servidor para poder realizar las funciones mencionadas en puntos anteriores.

Además, aunque en menor medida, he contribuido aportando ideas y ayudando a solucionar problemas del resto de la aplicación.

BIBLIOGRAFÍA

[1] The HL7 Clinical Document Architecture

[2] About DICOM: Overview

[3] About openEHR

[4] HL7 FHIR

[5] Visual Studio Code

[6] Lenguaje HTML. Qué es HTML

[7] Lenguaje CSS. Qué es CSS

[8] Bootstrap. Qué es Bootstrap

[9] MDN. Qué es Javascript

[10] Node.js. Acerca de Node.js

[11] Lenguaje JS. Qué es NPM

[12] MDN. Iniciación a Express/Node

[13] Hostinger. Qué es AJAX y cómo funciona

[14] Xataka. Qué es Github y qué es lo que ofrece a los desarrolladores

[15] IBM RSAD. What is IBM RSAD?

[16] Medium.com. Los 10 patrones comunes de arquitectura software

Anexo A - Guía de uso

En este capítulo se explica cómo hacer uso de la aplicación. Desde la ventana principal, tal y como se ve en la *Figura 22*, tenemos acceso a tres funcionalidades:

- Crear medidas propias
- Crear un modelo propio
- Adaptar un modelo externo

Figura 22- Captura Inicio

Si es la primera vez que se usa la aplicación, es necesario acceder en ese orden a cada ventana para poder proceder con la transformación de la base de datos externa.

A.1. Crear medidas propias

En esta ventana se crean las métricas que se usan para complementar tanto a los elementos del esquema como los del modelo de la base de datos externa.

Como se ve en la *Figura 23*, para crear una nueva métrica se pide el "tipo", que representa el nombre de la nueva métrica. Al hacer clic en "Añadir medida", se crean dos nuevas entradas. En la nueva entrada de la izquierda, "Medida", se escribe el nombre de una unidad de medida, mientras que en "Equivalencia", a la derecha, debe

ir una operación aritmético lógica, ya que es la operación que se hará durante la transformación de la base de datos externa para calcular los elementos con valores que no sean de tipo "Texto".

Al hacer clic en guardar, se escribe la nueva métrica al final del archivo que los almacena.

Figura 23- Captura Crear medidas propias

A.2. Crear un modelo propio

Se utiliza para crear el esquema que usaremos durante la transformación del modelo. Hay un elemento vacío al cargar la página, y según la *Figura 24* cada elemento contiene tres campos: el nombre, su métrica y su unidad de medida.

Al hacer clic en el botón "Añadir campo", se crea un nuevo elemento con los mismos campos vacíos.

Una vez se han creado todos los elementos y rellenado todos los campos, la información se guarda el esquema haciendo clic en el botón "Guardar", sobreescribiendo la información si ya existiera.

A.2.1. Establecer la métrica y unidad de un elemento

El selector de la métrica de un elemento empieza con una opción por defecto y el selector de la unidades de medida vacío.

Al seleccionar una métrica, el selector de las unidades de medida se actualiza con las opciones correspondientes. Si se selecciona la opción por defecto en el selector de la métrica otra vez, el selector de unidades de medida se vacía.

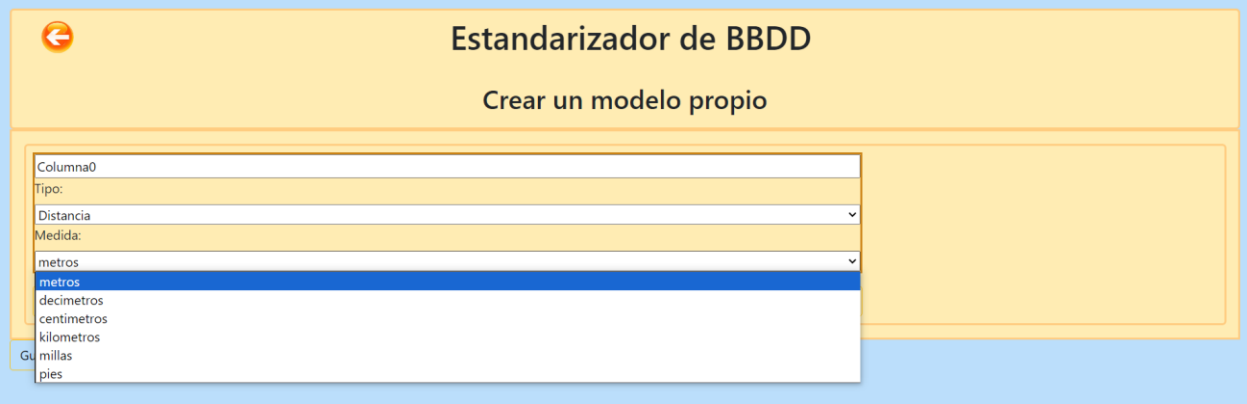

Figura 24- Captura Crear modelo propio

A.3. Adaptar un modelo externo

A.3.1. Cargar un modelo externo

El primer paso es hacer clic en "Seleccionar archivo" y seleccionar el modelo de la base de datos que queremos adaptar. Como el esquema, el modelo no contiene los datos de las filas de la base de datos, sino dos columnas: el nombre de las columnas de la base de datos externa en lenguaje natural y el nombre real de las columnas, separados por comas.

Como se aprecia en la *Figura 25*, una vez se carga el modelo en la mitad izquierda, se carga en la mitad derecha el esquema, y los conjuntos de relaciones que se hubieran creado en anteriores usos de la aplicación aparecen en el selector.

Figura 25- Captura Cargar un modelo externo

A.3.2. Seleccionar o crear una relación

Como muestra la *Figura* 26, el selector, una vez cargado, tiene tantas opciones como conjuntos de relaciones ya creados se hayan encontrado, así como dos opciones por defecto. La primera, "---", se selecciona por defecto y no permite vincular filas. La segunda, "Create New", aparecerá siempre al final de la lista y sirve para crear nuevos conjuntos de relaciones.

Al seleccionarlo, se limita el uso del resto de la página y aparece un cuadro para introducir el nombre del nuevo conjunto. Al cancelar, volvemos a la opción por defecto y, al aceptar, se crea un nuevo conjunto sin ninguna fila relacionada.

Si en su lugar selecciona un conjunto de relaciones existentes, se borran todas las relaciones temporales entre filas y se cargan las que se especifican en el conjunto cargado. El modelo debe tener tantas filas como se indica en el conjunto de relaciones para poder cargarlo.

Figura 26- Captura Establecer una relación

A.3.3. Establecer las métricas y unidades de medida de las filas

Como se puede ver en la *Figura 27*, una vez cargadas las filas, se pueden seleccionar las métricas y unidades de medida de los elementos del modelo externo en los selectores izquierdo y derecho, respectivamente. La opción por defecto es "Texto" y seleccionar otra opción carga las nuevas opciones con las unidades pertinentes en el selector derecho.

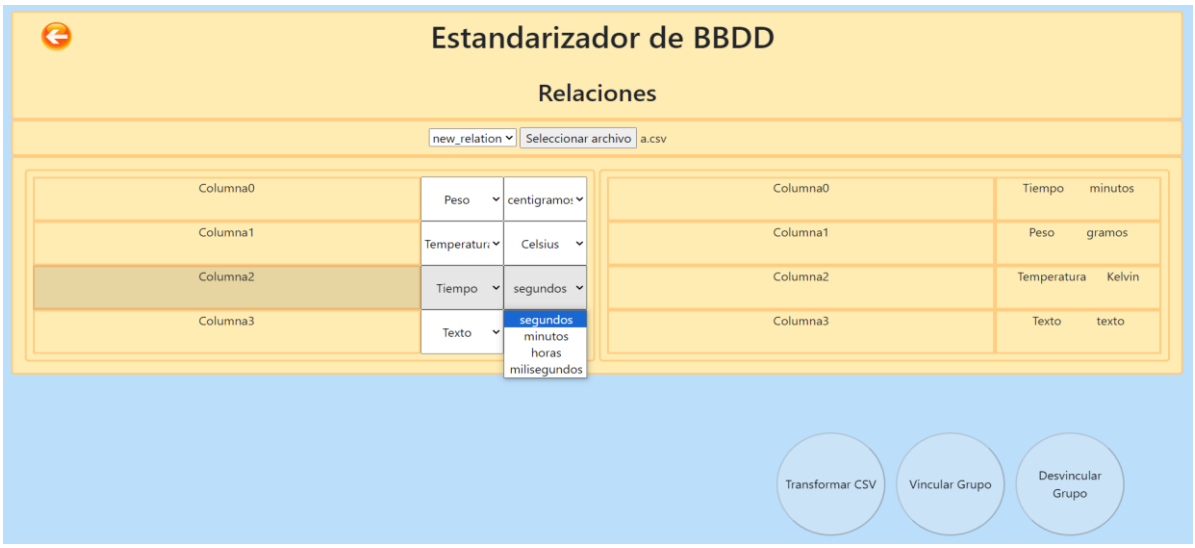

Figura 27- Captura Establecer métricas y medidas

A.3.4. Vincular y desvincular filas

Se puede hacer clic en las filas para resaltarlas, quedando seleccionadas para posibles acciones. Se pueden seleccionar tantas filas del modelo externo como se quiera, pero solo uno del derecho, que actúa como columna destino en las relaciones.

Una vez seleccionadas al menos una fila del modelo y una fila del esquema, se habilita el botón "Vincular" para establecer una relación entre ellas. Las métricas que no sean "Texto" deben coincidir, de lo contrario no se pueden vincular.

Las filas que ya están en una relación están resaltadas con un borde negro, tal y como muestra la *Figura 28*. Si se selecciona una de estas filas, en lugar de seleccionarse como en el caso anterior, todas las filas de la relación a la que pertenece quedan resaltadas con bordes blancos y se habilita el botón "Desvincular". Hacer clic en éste eliminará la relación y hará las filas afectadas seleccionables de nuevo para vincularlas si se quisiera, mientras que hacer clic en una fila ya resaltada o en otra fila cualquiera que no forme parte de la relación cancela la selección. En ambos casos se deshabilita el botón "Desvincular" al final.

Figura 28- Captura Vincular/desvincular

A.3.5. Transformar la base de datos externa

Una vez se hayan relacionado todas las filas que se deseen usar para la transformación de la base de datos externa, se hace clic en el botón "Transformar" para proceder con la adaptación de está al modelo. El botón de transformar se habilita si hay al menos una relación existente, y no es necesario que todas las filas del modelo externo y/o del esquema estén relacionadas. Las columnas de la base de datos resultante cuyas correspondientes filas del modelo no estén en una relación aparecerán vacías, y los datos de las columnas de la base de datos externa cuya fila correspondiente en su modelo no estén en una relación no se usarán.

A.3.5.1. Reordenar elementos

Como se puede ver en la *Figura 29*, al hacer clic en "Transformar", aparecerá un cuadro con la vista previa de las relaciones finales. Hacer clic en "Cancelar" cierra la vista previa y no se transforma la base de datos. Al hacer clic en "Confirmar" se procede con la transformación.

Antes de confirmar, debajo de estos botones, aparece la vista previa de la transformación, indicando qué columnas de la base de datos externa se van a juntar y en qué columna destino del esquema se guarda el resultado. Los elementos que aparecen a la izquierda del todo corresponden con las filas del modelo cuya métrica es "Texto", al lado de estas las filas del modelo con otras métricas y, a la derecha del todo, la columna destino.

Los elementos de la izquierda del todo se pueden reordenar. Para ello, se hace clic en un elemento válido, lo que lo resalta con un borde blanco, y se hace clic en otro elemento del mismo grupo en la misma fila, finalmente intercambiando sus posiciones.

Figura 29- Captura Reordenar elementos

A.3.5.2. Drag and Drop

Una vez se hace clic en confirmar, la vista previa se sustituye por un cuadro con funcionalidad Drag and Drop, como muestra la *Figura 30*. En este cuadro es donde podemos arrastrar y soltar las base de datos externa para finalmente transformarla de acuerdo a las relaciones creadas y el esquema.

En el proceso de transformación, para cada fila de la base de datos externa y por cada relación, se concatena los valores de las columnas con métricas de tipo "Texto", mientras que los valores de las columnas con métricas de otro tipo se transforman en la unidad de medida por defecto de acuerdo al archivo de medidas (eg kilómetros a metros, kelvin a celsius), se suman, y se transforman en la unidad de medida correspondiente de la columna destino. Si la columna destino fuera texto, no se realiza ninguna transformación más y se concatenan ambas partes.

Una vez este proceso finaliza, se descarga un archivo en formato csv con la base de datos adaptada al esquema y las transformaciones pertinentes.

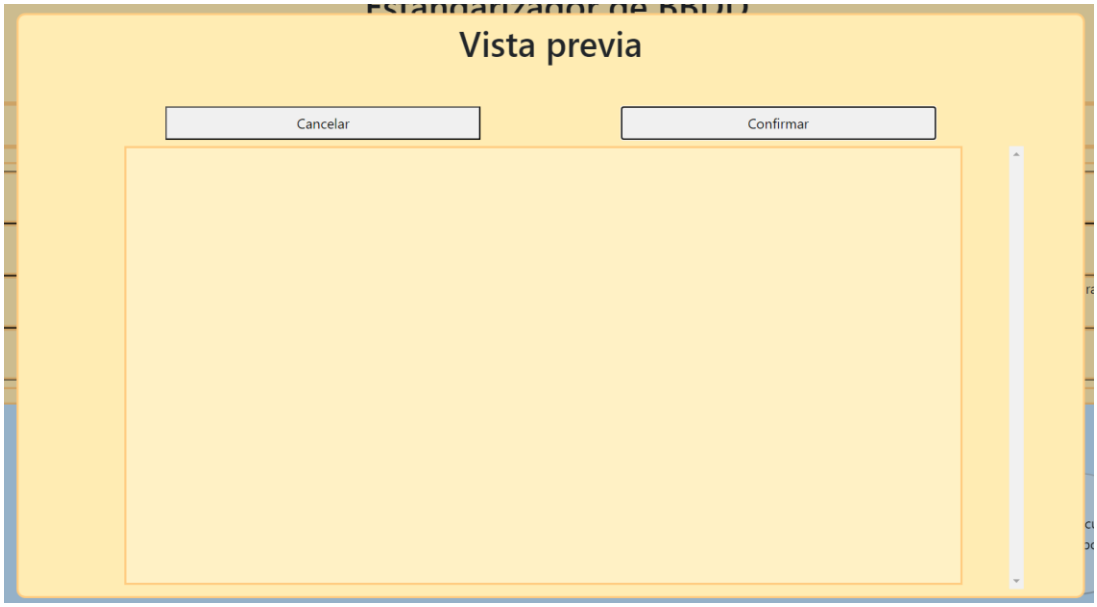

Figura 30- Captura Drag and drop