



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**Draw E-R App
Documentación Técnica**



Presentado por Rubén Maté Iturriaga
en Universidad de Burgos — 7 de julio de 2024
Tutor: Jesús Manuel Maudes Raedo

Índice general

Índice general	i
Índice de figuras	iii
Índice de tablas	v
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal	1
A.3. Estudio de viabilidad	10
Apéndice B Especificación de Requisitos	15
B.1. Introducción	15
B.2. Objetivos generales	15
B.3. Catálogo de requisitos	16
B.4. Especificación de requisitos	18
Apéndice C Especificación de diseño	25
C.1. Introducción	25
C.2. Diseño de datos	25
C.3. Diseño procedimental	27
C.4. Diseño arquitectónico	30
Apéndice D Documentación técnica de programación	33
D.1. Introducción	33
D.2. Estructura de directorios	33
D.3. Manual del programador	34

D.4. Compilación, instalación y ejecución del proyecto	37
D.5. Pruebas del sistema	39
Apéndice E Documentación de usuario	41
E.1. Introducción	41
E.2. Requisitos de usuarios	41
E.3. Instalación	41
E.4. Manual del usuario	43
Apéndice F Anexo de sostenibilización curricular	53
F.1. Introducción	53
F.2. Competencias de sostenibilidad adquiridas	53
F.3. Reflexión final	55
Bibliografía	57

Índice de figuras

A.1. División en sprints del desarrollo	2
A.2. Labels a asignar a los diferentes issues y pull requests	3
A.3. Issues del Sprint 0	4
A.4. Issues del Sprint 1	5
A.5. Issues del Sprint 2	6
A.6. Issues del Sprint 3	7
A.7. Issues del Sprint 4 (Parte 1)	8
A.8. Issues del Sprint 4 (Parte 2)	9
A.9. Régimen general de la Seguridad Social	11
C.1. Representación interna del diagrama E-R	26
C.2. Diagrama E-R que representa la estructura interna sobre la que funciona la aplicación.	27
C.3. Diagrama de flujo que representa la funcionalidad de mover objetos en el canvas de la aplicación.	28
C.4. Diagrama de flujo que representa la funcionalidad de generar el script SQL en la aplicación.	29
C.5. Diagrama con las partes de la aplicación en su ejecución en el navegador.	31
D.1. Ejecución de las diferentes GitHub Actions en nuestro proyecto.	37
D.2. Ejecución de pruebas unitarias	39
D.3. Ejecución de pruebas End to End con la interfaz gráfica de Playwright	40
E.1. Pantalla inicial de la aplicación.	43
E.2. Diferentes componentes de la aplicación.	43
E.3. Acciones contextuales sobre una entidad.	45

E.4. Acciones contextuales sobre un atributo.	46
E.5. Acciones contextuales sobre una relación.	47
E.6. Añadir atributos en una relación N-M.	47
E.7. Diálogo para configurar relaciones.	48
E.8. Diálogo para configurar cardinalidades de una relación.	49
E.9. Diagnóstico al generar SQL con un diagrama no válido.	50
E.10. Diálogo con selección de archivo para import diagrama JSON. . .	50
E.11. Diálogo de confirmación para reiniciar el canvas.	51

Índice de tablas

A.1. Issues completadas Sprint 0	4
A.2. Issues completadas Sprint 1	5
A.3. Issues completadas Sprint 2	6
A.4. Issues completadas Sprint 3	7
A.5. Issues completadas Sprint 4	9
A.6. Porcentajes de impuestos aplicables	10
A.7. Costes totales	13
A.8. Tabla de licencias	13
B.1. CU-1 Modelar diagramas E-R.	19
B.2. CU-2 Almacenar diagramas.	20
B.3. CU-3 Validar diagramas.	21
B.4. CU-4 Exportar diagramas.	22
B.5. CU-5 Importar diagramas.	23
B.6. CU-6 Generar SQL del diagrama modelado.	24

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En este apartado se realiza un pequeño resumen del desarrollo del proyecto y de los diferentes sprints que lo han compuesto, así como un estudio de viabilidad en el que se presenta tanto la viabilidad económica como legal del proyecto.

A.2. Planificación temporal

Para el desarrollo de este trabajo de fin de grado se ha usado una metodología de desarrollo ágil. Los sprints han sido de aproximadamente unas 3-4 semanas, lo cuál ha permitido poder compaginarlo con mi trabajo. Se han administrado con las Milestones de la plataforma GitHub en combinación con las labels que determinan el tipo de tarea que es cada issue.

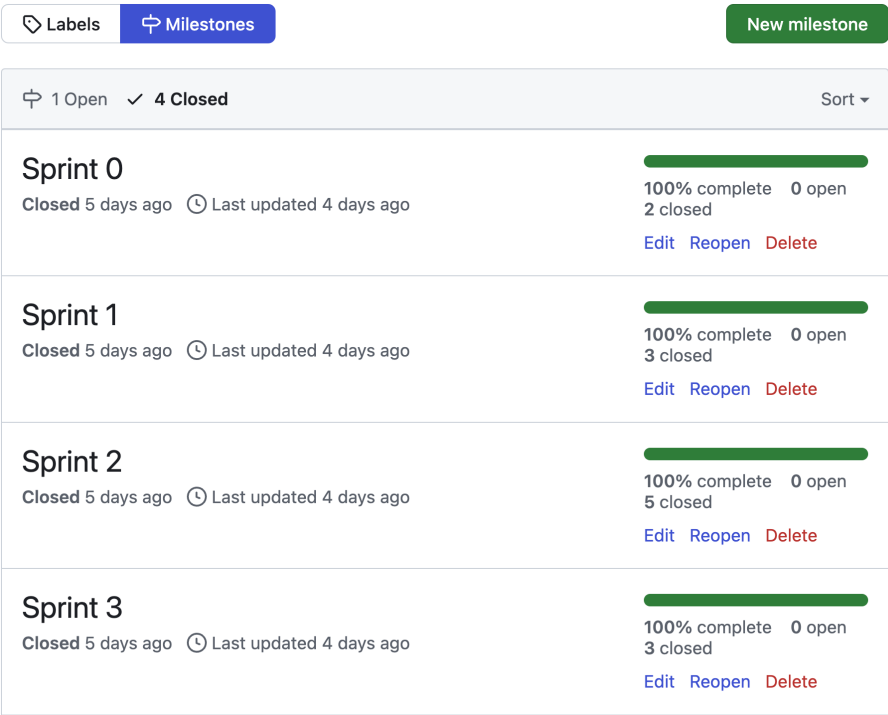


Figura A.1: División en sprints del desarrollo

LabelsMilestones

Search all labels

New label

7 labels		Sort
bug	Something isn't working	
documentation	Improvements or additions to documentation	1
enhancement	New feature or request	
important	Important issue that should be top priority	
research	Investigation required	
testing	Testing related issued	
tooling	Additions to the app tooling	

Figura A.2: Labels a asignar a los diferentes issues y pull requests

Sprint 0 - 13/03/2024 - 8/04/2024

El primer sprint ha consistido en plantear las bases sobre las que se iba a cimentar el proyecto. Esto es:

- Decidir el framework tecnológico a usar. El debate estaba entre React [4] o Javascript [14], se optó finalmente por este primero puesto que no se le ve un gran sentido a desarrollar puramente con Javascript a estas alturas del desarrollo web.
- Decidir la librería para dibujar/modelar: Se optó finalmente por mxGraph[7]/maxGraph[8].

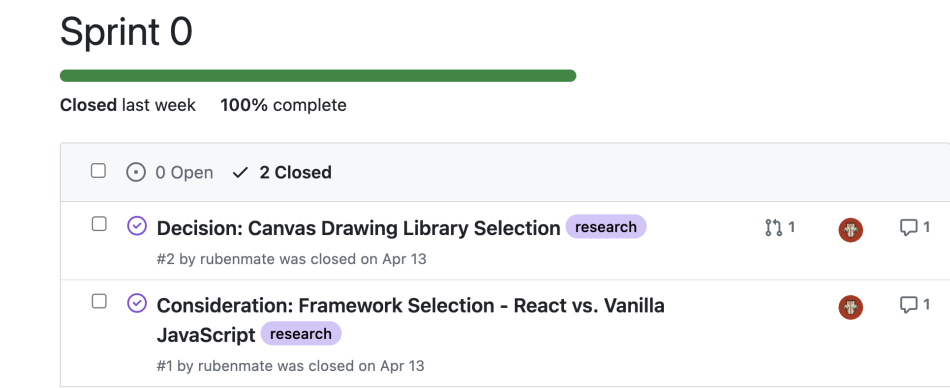


Figura A.3: Issues del Sprint 0

Issues	Label
Decision: Canvas Drawing Library Selection	research
Consideration: Framework Selection - React vs Vanilla Javascript	research

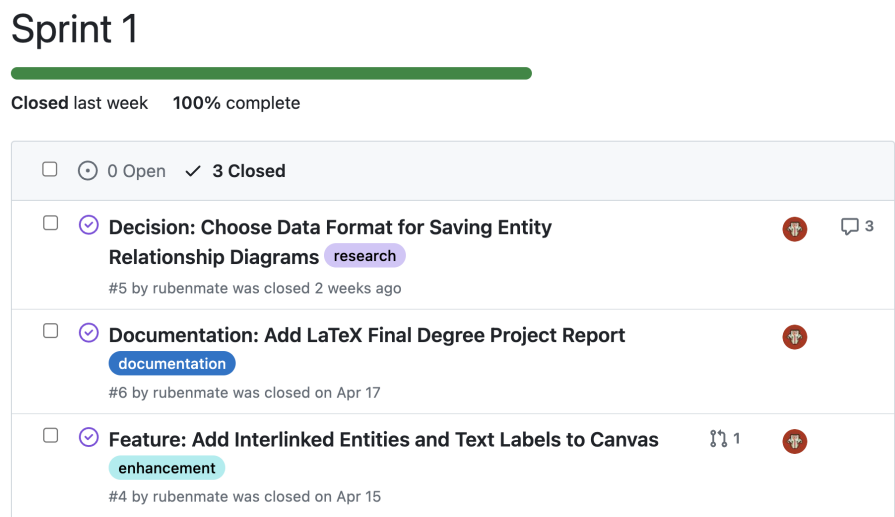
Tabla A.1: Issues completadas Sprint 0

Sprint 1 - 9/04/2024 - 24/04/2024

El segundo sprint ha consistido en:

- Decidir qué formato de datos va a usarse para guardar y poder tratar un diagrama E-R. Este issue no se completará totalmente hasta una fase casi final del proyecto.

- Añadir la plantilla de la memoria LaTeX al repositorio.
- Implementar la funcionalidad inicial de poder añadir Entidades y cambiar su nombre.



- Mostrar/ocultar atributos de las entidades con un menú contextual.
- Añadir una función de utilidad que compruebe si un determinado diagrama es válido. Esta función no se completará en su totalidad hasta un punto posterior.

Sprint 2

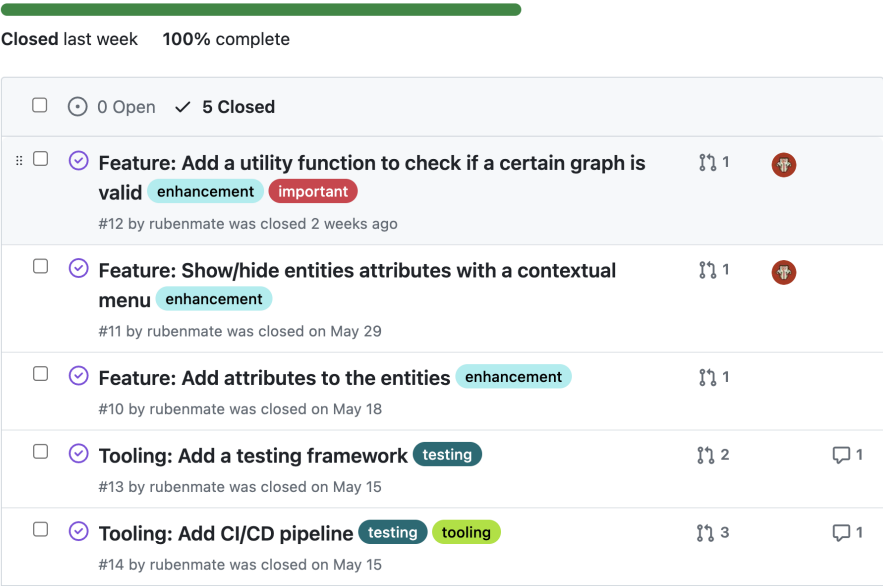


Figura A.5: Issues del Sprint 2

Issues	Label
Tooling: Add CI/CD pipeline	testing
Tooling: Add a testing framework	testing
Feature: Add attributes to the entities	enhancement
Feature: Show/hide entities attributes with a contextual menu	enhancement
Feature: Add a utility function to check if a certain graph is valid	enhancement

Tabla A.3: Issues completadas Sprint 2

Sprint 3 - 29/05/2024 - 15/06/2024

El cuarto sprint ha consistido en:

- Cambiar la iconografía alrededor de los atributos de las entidades para que el atributo clave tuviera su label subrayada.

- Cambiar el modo en que se añaden los atributos. Ya no se escoge si se quiere añadir clave, sino que el primer atributo añadido es clave y el resto no. Con un menú contextual se puede convertir no clave en clave.
- Añadir relaciones (simples) entre entidades.

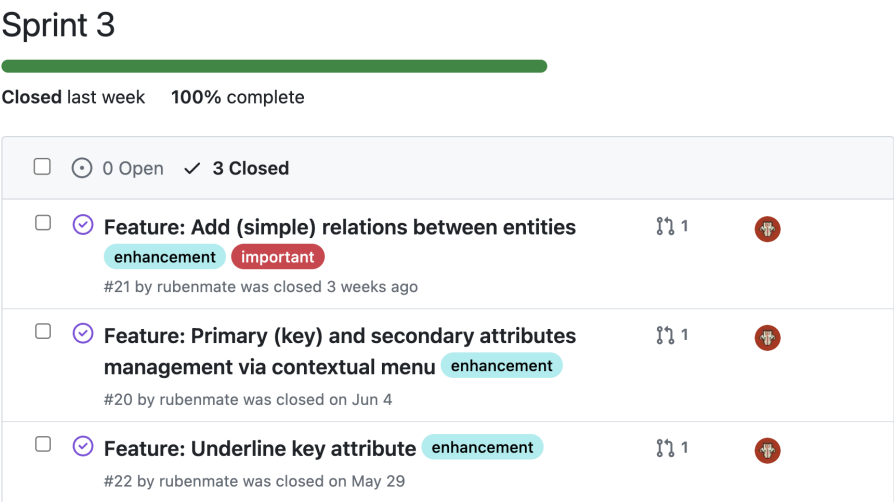


Figura A.6: Issues del Sprint 3

Issues	Label
Feature: Underline key attribute	enhancement
Feature: Primary (key) and secondary attributes management via contextual menu	enhancement
Feature: Add (simple) relations between entities	enhancement

Tabla A.4: Issues completadas Sprint 3

Sprint 4 - 15/06/2024 - 05/07/2024

El cuarto y último sprint ha consistido en:

- Implementar la funcionalidad para que las relaciones 'N:M' puedan tener atributos.
- Implementar la funcionalidad para prevenir que atributos, entidades y relaciones se inicien con nombres repetidos.

- Implementar la funcionalidad para generar el script SQL correspondiente al paso a tablas del diagrama modelado.
- Implementar la funcionalidad para añadir botones de Borrar en los diversos elementos.
- Implementar la funcionalidad para poder reconfigurar relaciones.
- Implementar la funcionalidad para que los atributos se muevan al mover la entidad y evitar que los nuevos atributos se inicialicen todos apilados.
- Implementar la funcionalidad para guardar el diagrama en el almacenamiento local del navegador y poder importar/exportar los diagramas en formato JSON.

Sprint 4

Closed 1 minute ago 100% complete

<input type="checkbox"/>	<input checked="" type="radio"/> 0 Open ✓ 8 Closed			
⋮ <input type="checkbox"/>	<input checked="" type="radio"/> Documentation documentation #45 by rubenmate was closed 1 hour ago <input type="radio"/> 4 tasks done	🔗 1	🛑	
<input type="checkbox"/>	<input checked="" type="radio"/> Feature: Save the diagram on localStorage, add export/import diagram enhancement #29 by rubenmate was closed 3 days ago	🔗 1	🛑	💬 1
<input type="checkbox"/>	<input checked="" type="radio"/> Feature: Better attributes drawing management enhancement #25 by rubenmate was closed last week	🔗 1	🛑	
<input type="checkbox"/>	<input checked="" type="radio"/> Feature: Reconfigure relations enhancement #41 by rubenmate was closed 2 weeks ago	🔗 1	🛑	

Figura A.7: Issues del Sprint 4 (Parte 1)

<input type="checkbox"/>	<input checked="" type="checkbox"/>	Feature: Add delete buttons for the different elements enhancement important #38 by rubenmate was closed 2 weeks ago 4 tasks done	1	
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Feature: Generate SQL script for the tables of the current diagram enhancement important #31 by rubenmate was closed 2 weeks ago	1	
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Feature: Prevent Duplicate Names in Attributes, Entities, and Relations enhancement #34 by rubenmate was closed 2 weeks ago	1	
<input type="checkbox"/>	<input checked="" type="checkbox"/>	Feature: 'N:M' relations can have attributes enhancement important #32 by rubenmate was closed 2 weeks ago	1	

Figura A.8: Issues del Sprint 4 (Parte 2)

Issues	Label
Feature: 'N:M' relations can have attributes	enhancement
Feature: Prevent Duplicate Names in Attributes, Entities and Relations	enhancement
Feature: Generate SQL script for the tables of the current diagram	enhancement
Feature: Add delete buttons for the different elements	enhancement
Feature: Reconfigure relations	enhancement
Feature: Better attributes drawing management	enhancement
Feature: Save the diagram on localhost, add export/import diagram	enhancement
Documentation	documentation

Tabla A.5: Issues completadas Sprint 4

A.3. Estudio de viabilidad

En esta sección se va a realizar un estudio de la viabilidad económica y legal del proyecto.

Viabilidad económica

Costes

En primer lugar, se detallan los costes asociados al proyecto.

■ Costes de personal

En este apartado se detalla el gasto asociado al personal involucrado en el proyecto. La estimación que se ha hecho de horas de trabajo es de 400 horas repartidas a lo largo de 4 meses, con un cálculo de 25 horas semanales. El salario del alumno se estima en 20 EUR/hora, lo que se traduce en un salario bruto mensual de:

$$25 \frac{\text{horas}}{\text{semana}} \times 20 \frac{\text{EUR}}{\text{hora}} \times 4 \frac{\text{semanas}}{\text{mes}} = 2000 \text{ EUR al mes}$$

Este cálculo corresponde al salario bruto del empleado.

Para determinar el salario neto que recibirá el empleado, se deben incluir los impuestos que la empresa debe pagar por él. Estos impuestos se han consultado y obtenido de la página oficial de la seguridad social: <http://www.seg-social.es/wps/portal/wss/internet/Trabajadores/CotizacionRecaudacionTrabajadores/36537>.

Tipo de impuesto	Porcentaje
Contingencias	23.6 %
Desempleo	5.5 %
FOGASA	0.20 %
Formación profesional	0.60 %

Tabla A.6: Porcentajes de impuestos aplicables

TIPOS DE COTIZACIÓN (%)				
CONTINGENCIAS	EMPRESA	TRABAJADORES	TOTAL	Accidentes de Trabajo y Enfermedades Profesionales
Comunes	23,60	4,70	28,30	Tarifa Primas establecida en la disposición adicional cuarta Ley 42/2006, de 28 de diciembre, PGE 2007, en la redacción dada por la Disposición Final Quinta del RDL 28/2018 de 28 de diciembre (BOE del 29) siendo las primas resultantes a cargo exclusivo de la empresa
Horas Extraordinarias Fuerza Mayor	12,00	2,00	14,00	
Resto Horas Extraordinarias	23,60	4,70	28,30	
Mecanismo Equidad Intergeneracional (MEI)	0,58	0,12	0,7	

DESEMPLEO				EMPRESA	TRABAJADORES	TOTAL
Tipo General:				5,50	1,55	7,05
Contratación indefinida, incluidos los contratos indefinidos a tiempo parcial y fijos discontinuos, contratación de duración determinada en las modalidades de contratos de formación en alternancia, para la formación y aprendizaje, formativo para la obtención de la práctica profesional adecuada al nivel de estudios, de relevo, interinidad y contratos realizados con trabajadores que tengan reconocido un grado de discapacidad no inferior al 33%						
Contrato duración determinada a tiempo completo o a tiempo parcial				6,70	1,60	8,30

	EMPRESA	TRABAJADORES	TOTAL
FOGASA	0,20		0,20

	EMPRESA	TRABAJADORES	TOTAL
FORMACIÓN PROFESIONAL	0,60	0,10	0,70

Figura A.9: Régimen general de la Seguridad Social

Teniendo en cuenta estos impuestos, el coste total del empleado es:

$$\frac{2000 \frac{\text{EUR}}{\text{mes}}}{1 - (0,236 + 0,055 + 0,002 + 0,006)} = 2,855,77 \text{ EUR al mes}$$

Además, se cuenta con un profesor contratado como apoyo, con un salario de 35 EUR/hora debido a su experiencia. El profesor trabaja tres horas a la semana:

$$3 \frac{\text{horas}}{\text{semana}} \times 35 \frac{\text{EUR}}{\text{hora}} \times 4 \frac{\text{semanas}}{\text{mes}} = 420 \text{ EUR al mes}$$

Sumando el profesor, se obtiene un total de 420 EUR brutos, a los cuales también se deben sumar los impuestos:

$$\frac{420 \frac{\text{EUR}}{\text{mes}}}{1 - (0,236 + 0,055 + 0,002 + 0,006)} = 599,18 \text{ EUR al mes para el profesor}$$

En resumen, la empresa deberá pagar mensualmente 3.454,95 EUR en total. Dado que el proyecto ha durado cuatro meses, el coste total asciende a **13.819,80 EUR**.

■ Hardware

Los recursos de hardware utilizados han sido un ordenador portátil MacBook Air cuyo coste fue de 1250 EUR y que ha sido amortizado completamente en ejercicios anteriores. Por lo tanto, podemos concluir que los costes de hardware han sido de **0 EUR**.

■ Software

En el proyecto, se han utilizado diversas herramientas de software, la mayoría de las cuales son gratuitas. Estas herramientas incluyen:

- **neovim**: Editor de texto gratuito.
- **Navegador web**: Herramienta gratuita para la navegación y pruebas web.
- **CleanShot X**: Herramienta para realizar capturas de pantalla, cuyo coste ha sido de 20 EUR como pago único.

Por lo tanto, el coste total del software utilizado en el proyecto ha sido de **20 EUR**.

■ Costes indirectos

En este apartado se incluyen los costes indirectos asociados al proyecto. Se ha utilizado una tarifa de internet por la que se está pagando 20 EUR mensuales. Dado que el proyecto ha durado cuatro meses, el coste total asciende a:

$$20 \text{ EUR/mes} \times 4 \text{ meses} = 80 \text{ EUR}$$

■ Total

La tabla de costes totales se detalla a continuación: [A.7](#)

Beneficios

Este es un proyecto con carácter educativo, por lo que no se espera obtener beneficios económicos directos.

Tipo de costes	Total
<i>Personal</i>	13.819,80 EUR
<i>Hardware</i>	0 EUR
<i>Software</i>	20 EUR
<i>Costes indirectos</i>	80 EUR
Total	13.919,80 EUR

Tabla A.7: Costes totales

Viabilidad legal

A continuación podemos observar en una tabla las licencias que tienen las diferentes dependencias que se encuentran en nuestra aplicación.

Herramienta/Librería	Versión	Licencia
React	17.0.0	MIT
mxGraph	4.1.0	Apache License 2.0
Material UI	5.15.19	MIT
Biome	1.7.3	MIT
Playwright	1.44.0	Apache License 2.0
Lefthook	1.6.11	MIT
Vitest	1.6.0	MIT

Tabla A.8: Tabla de licencias

Las herramientas y librerías utilizadas en este proyecto están cubiertas por dos tipos exclusivamente: la licencia MIT[10] y la licencia Apache 2.0[1]. Ambas son permisivas y permiten la reutilización, modificación y distribución del software.

La licencia MIT es muy permisiva y permite a los usuarios utilizar, copiar, modificar, fusionar, publicar, distribuir, sublicenciar y/o vender copias del software [11]. Siempre que se incluya esta licencia en todas las copias o partes sustanciales del software.

La licencia Apache 2.0 es también muy permisiva a la hora de utilizar, distribuir y modificar versiones del software. También, protege a los desarrolladores de ser responsables de los daños ocasionados [13].

Por lo tanto, la licencia de nuestro proyecto puede ser licencia MIT, ya que todas las herramientas y librerías utilizadas permiten su uso bajo estas condiciones. Esto facilita la integración y distribución del proyecto, asegurando que cumpla con las licencias de todas las dependencias utilizadas.

Apéndice B

Especificación de Requisitos

B.1. Introducción

En este apartado se enumeran y desarrollan los objetivos y requisitos que debe tener la aplicación según han sido marcados al comienzo del proyecto.

B.2. Objetivos generales

Este proyecto tiene como objetivo los siguientes puntos:

- Modelar diagramas E-R.
- Almacenar los diagramas en una estructura interna fácilmente extensible.
- Validar esa estructura interna para conocer si un diagrama E-R es válido.
- Exportar un diagrama válido a JSON.
- Importar y recrear un diagrama en JSON válido.
- Exportar los diagramas válidos a scripts SQL.

B.3. Catálogo de requisitos

En este apartado se definirán los requisitos funcionales y no funcionales del proyecto.

Requisitos funcionales

- **RF1 - Modelar diagramas E-R:** La aplicación debe permitir a los usuarios crear y manipular diagramas E-R.
- **RF2 - Almacenar diagramas:** La aplicación debe almacenar los diagramas en una estructura interna que sea fácilmente extensible.
- **RF3 - Validar diagramas:** La aplicación debe validar la estructura interna del diagrama para asegurar su corrección al generar el SQL o exportar/importar JSON.
- **RF4 - Exportar diagramas:** La aplicación debe permitir exportar los diagramas validados a un archivo JSON.
- **RF5 - Importar diagramas:** La aplicación debe permitir importar diagramas desde un archivo JSON y recrearlos en el entorno de la aplicación.
- **RF6 - Generar SQL:** La aplicación debe permitir generar scripts SQL a partir de los diagramas validados.

Requisitos no funcionales

- **RNF1 - Usabilidad:** La aplicación debe ser fácil de usar y tener una interfaz intuitiva para mejorar la experiencia del usuario.
- **RNF2 - Rendimiento:** La aplicación debe ser eficiente en términos de rendimiento.
- **RNF3 - Compatibilidad:** La aplicación debe ser compatible con diferentes navegadores.
- **RNF4 - Seguridad:** La aplicación debe garantizar la seguridad de los datos manejados y almacenados.
- **RNF5 - Mantenibilidad:** La aplicación debe ser fácil de mantener y actualizar. Debe poder expandirse en el futuro con nuevos tipos de relaciones y funcionalidades.

B.4. Especificación de requisitos

Casos de uso

CU-1	Modelar diagramas E-R
Versión	1.0
Autor	Rubén Maté Iturriaga
Requisitos asociados	RF-1
Descripción	Permite al usuario crear y manipular diagramas E-R.
Precondición	La aplicación debe estar abierta y funcionando.
Acciones	<ol style="list-style-type: none">1. El usuario arrastra y suelta elementos (entidades, relaciones) al canvas.2. El usuario añade atributos a las entidades y relaciones (en caso de que fueran N:M).3. El usuario edita los nombres y atributos de las entidades y relaciones.4. El usuario vincula entidades configurando las relaciones y asignándolas cardinalidades.
Postcondición	El diagrama se almacena en la estructura interna de la aplicación.
Excepciones	-
Importancia	Alta

Tabla B.1: CU-1 Modelar diagramas E-R.

CU-2	Almacenar diagramas
Versión	1.0
Autor	Rubén Maté Iturriaga
Requisitos asociados	RF-2
Descripción	Permite almacenar los diagramas creados en una estructura interna.
Precondición	El usuario debe haber creado un diagrama E-R.
Acciones	<ol style="list-style-type: none">1. La estructura interna se mantiene actualizada automáticamente ante los diversos cambios y acciones.
Postcondición	El diagrama se almacena correctamente y puede ser recuperado.
Excepciones	Problemas de almacenamiento (mensaje de error).
Importancia	Alta

Tabla B.2: CU-2 Almacenar diagramas.

CU-3	Validar diagramas
Versión	1.0
Autor	Rubén Maté Iturriaga
Requisitos asociados	RF-3
Descripción	Permite validar la estructura interna del diagrama para asegurar su corrección al generar el SQL o exportar/importar JSON.
Precondición	El usuario debe haber creado y modelado un diagrama.
Acciones	<ol style="list-style-type: none">1. El usuario selecciona la opción de generar SQL o exportar/importar diagrama.2. La aplicación verifica la corrección del diagrama según las reglas E-R.
Postcondición	La aplicación muestra un mensaje indicando si el diagrama es válido o no antes de generar el SQL o exportar/importar JSON.
Excepciones	Mensajes de error con diagnóstico en caso de diagrama no válido.
Importancia	Alta

Tabla B.3: CU-3 Validar diagramas.

CU-4	Exportar diagramas
Versión	1.0
Autor	Rubén Maté Iturriaga
Requisitos asociados	RF-4
Descripción	Permite exportar los diagramas válidos a un archivo JSON.
Precondición	El diagrama debe ser válido.
Acciones	<ol style="list-style-type: none"> 1. El usuario selecciona la opción de exportar diagrama. 2. La aplicación genera un archivo JSON con la estructura interna del diagrama. 3. El usuario descarga (la descarga se realiza automáticamente al pulsar el botón Aceptar) el archivo JSON.
Postcondición	El archivo JSON se guarda en el dispositivo del usuario.
Excepciones	Problemas de exportación (mensaje de error).
Importancia	Media

Tabla B.4: CU-4 Exportar diagramas.

CU-5	Importar diagramas
Versión	1.0
Autor	Rubén Maté Iturriaga
Requisitos asociados	RF-5
Descripción	Permite importar diagramas desde un archivo JSON y recrearlos en la aplicación.
Precondición	El usuario debe tener un archivo JSON válido.
Acciones	<ol style="list-style-type: none">1. El usuario selecciona la opción de importar diagrama.2. El usuario carga el archivo JSON en la aplicación.3. La aplicación hace una validación del diagrama importado y si es correcta recrea el diagrama.
Postcondición	El diagrama se visualiza correctamente en la aplicación.
Excepciones	Problemas de importación (mensaje de error).
Importancia	Media

Tabla B.5: CU-5 Importar diagramas.

CU-6	Generar SQL del diagrama modelado
Versión	1.0
Autor	Rubén Maté Iturriaga
Requisitos asociados	RF-6
Descripción	Permite generar un script SQL a partir del diagrama E-R modelado y validado.
Precondición	El diagrama debe ser válido.
Acciones	<ol style="list-style-type: none"> 1. El usuario selecciona la opción de generar SQL. 2. La aplicación valida el diagrama E-R. 3. La aplicación genera un script SQL con la estructura del diagrama. 4. El usuario descarga el archivo SQL.
Postcondición	El archivo SQL se guarda en el dispositivo del usuario.
Excepciones	Problemas de generación de SQL (mensaje de error).
Importancia	Media

Tabla B.6: CU-6 Generar SQL del diagrama modelado.

Apéndice C

Especificación de diseño

C.1. Introducción

En este apéndice se detallan las decisiones de diseño tomadas durante el desarrollo. Se cubren los aspectos relacionados con el diseño de datos, el diseño procedimental y el diseño arquitectónico.

C.2. Diseño de datos

Esta aplicación no tiene como tal una estructura de datos definida puesto que no cuenta con un backend ni base de datos. Sin embargo sí que tiene la estructura JSON interna que contiene nuestro diagrama.

En esta estructura tenemos 3 entidades definidas que son: Entidades, Relaciones y Atributos.

Su diagrama E-R es el siguiente: [C.2](#)

```

{
  "entities": [
    {
      "idMx": "2",
      "name": "Entidad",
      "position": {
        "x": 254,
        "y": 130
      },
      "attributes": [
        {
          "idMx": "3",
          "name": "Atributo",
          "position": {
            "x": 374,
            "y": 130
          }
        }
      ]
    }
  ],
  "relations": [
    {
      "idMx": "8",
      "name": "Relacion",
      "position": {
        "x": 215,
        "y": 110
      },
      "side1": {
        "idMx": "11",
        "cardinality": "0:1",
        "cell": "11",
        "entity": {
          "idMx": "2"
        }
      },
      "side2": {
        "idMx": "12",
        "cardinality": "1:N",
        "cell": "12",
        "entity": {
          "idMx": "2"
        }
      },
      "canHoldAttributes": false,
      "attributes": []
    }
  ]
}

```

Figura C.1: Representación interna del diagrama E-R

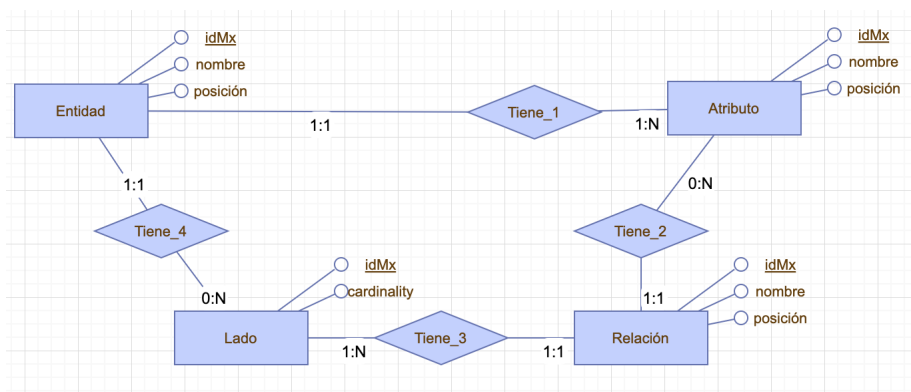


Figura C.2: Diagrama E-R que representa la estructura interna sobre la que funciona la aplicación.

C.3. Diseño procedimental

En esta sección se describen los procesos más complejos de la aplicación utilizando diagramas de flujo para entenderlos más fácilmente.

En general ninguno de los procesos de la aplicación requiere grandes diagramas de flujo para comprenderlos. Sin embargo, sí que hay dos funcionalidades que ameritan el tener sus propios diagramas de flujo.

Mover objetos

Es una de las partes más complejas puesto que mover objetos ha de activar diversas tareas según el tipo de objeto que se esté moviendo. Los objetos pueden ser:

- Entidades.
- Relaciones.
 - Reflexivas.
 - No reflexivas.
- Atributos.

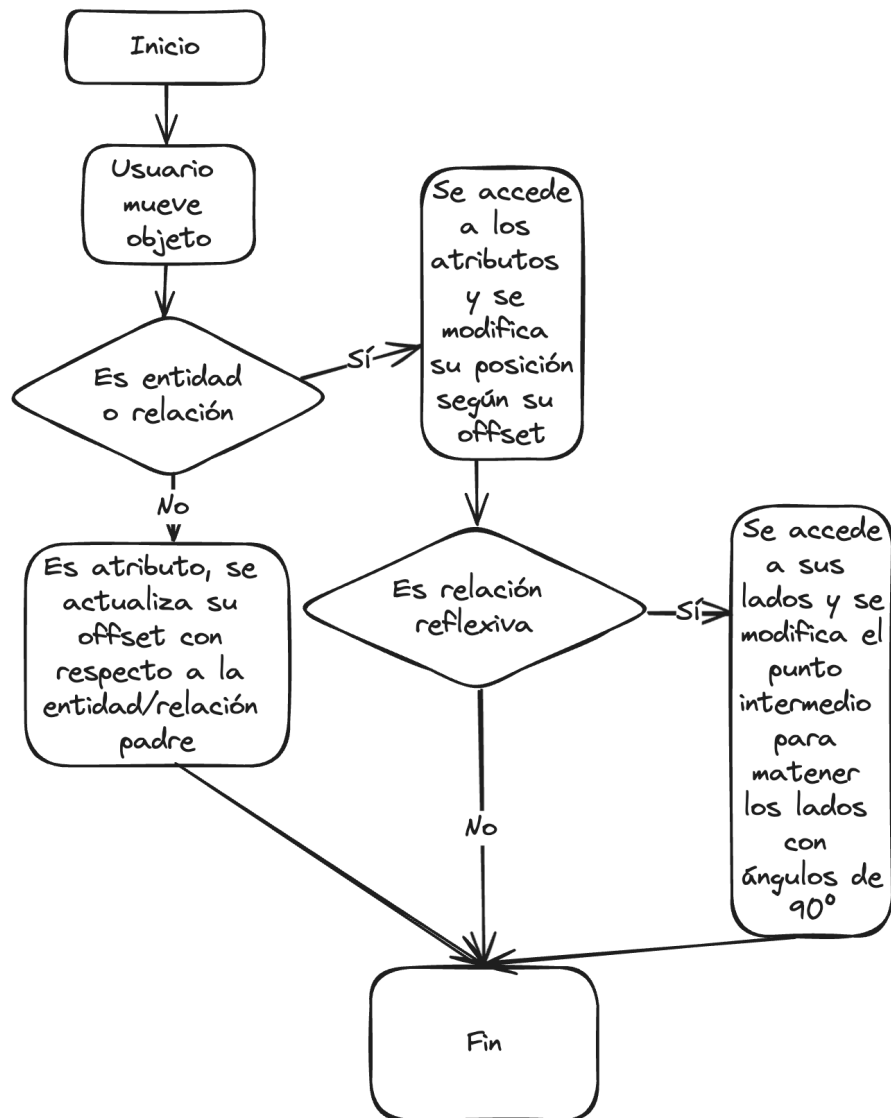


Figura C.3: Diagrama de flujo que representa la funcionalidad de mover objetos en el canvas de la aplicación.

Generar el script SQL

El script SQL es una de las funcionalidades más complejas de la aplicación. Requiere, aparte de la validación, un gran número de pasos y tablas intermedias donde se procesan los diferentes tipos de relaciones.

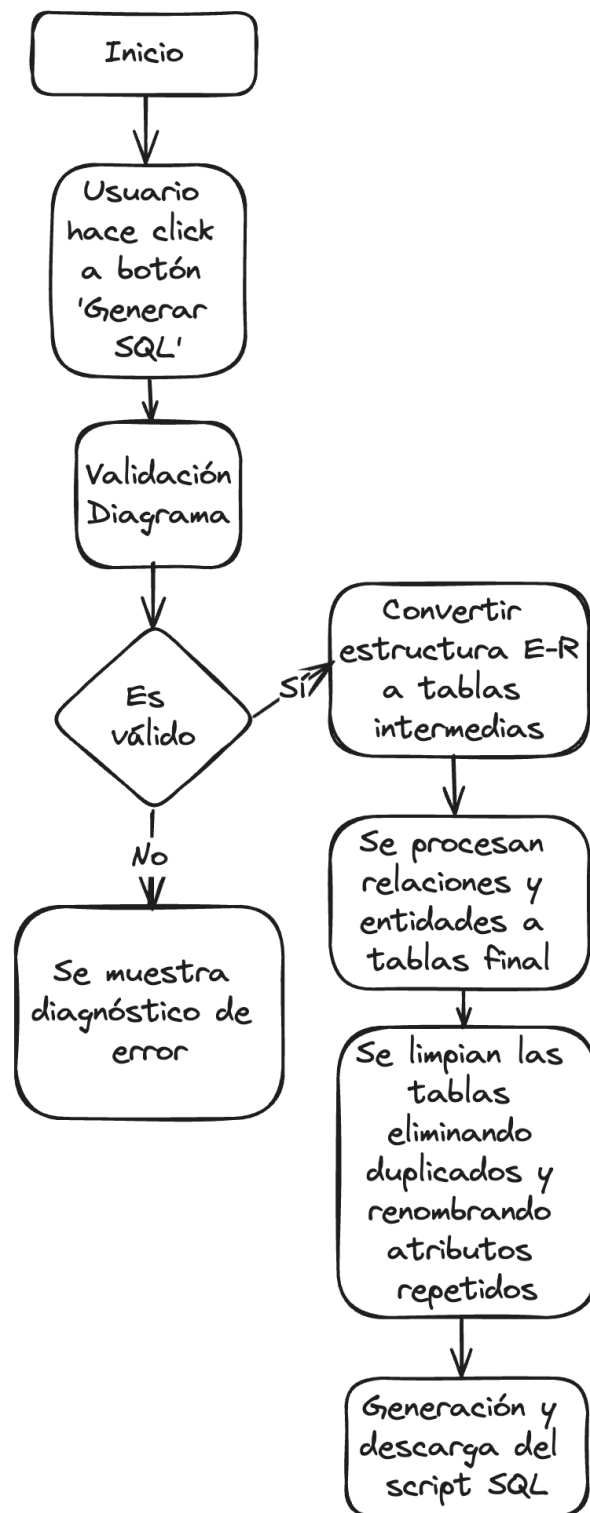


Figura C.4: Diagrama de flujo que representa la funcionalidad de generar el script SQL en la aplicación.

C.4. Diseño arquitectónico

El diseño arquitectónico de la aplicación es bastante simple, puesto que se basa en una arquitectura frontend utilizando React y la librería `mxGraph` para el modelado de los diagramas E-R.

Estructura de paquetes

La estructura de paquetes de la aplicación se organiza de la siguiente manera:

- **src:** Contiene todo el código fuente de la aplicación.
 - **components/DiagramEditor**
 - **styles:** Carpeta con el css que afecta al componente `DiagramEditor`
 - **DiagramEditor.js:** Componente principal de nuestra aplicación, llama al resto de módulos e inicializa la librería `mxGraph`.
 - **utils**
 - ◊ **setInitialConfiguration.js:** Ajusta la configuración la que se inicializa la librería `mxGraph`.
 - ◊ **initToolbar.js:** Inicializa la barra de herramientas.
 - ◊ **addToolbarItem.js:** Función auxiliar que se llama al arrastrar un componente de la barra de tareas al canvas.
 - ◊ **configureToolbar.js:** Función auxiliar para configurar la barra de herramientas.
 - ◊ **getStyleByKey.js:** Función auxiliar para obtener estilos de un objeto.
 - ◊ **getStyleStringByObj.js:** Función auxiliar para obtener estilos de un objeto.
 - **utils:**
 - **validation.js:** Módulo que contiene las funciones necesarias para procesar y determina si la estructura interna E-R conforma un diagrama válido.
 - **sql.js:** Módulo que contiene las funciones necesarias para procesar y convertir la estructura interna E-R en un script SQL.

Componentes de la arquitectura

Aunque la aplicación es puramente frontend, sí se pueden apreciar ciertos componentes externos a la aplicación:

- **Archivos JSON:** Utilizados para la persistencia de los diagramas. Pueden ser importados.
- **LocalStorage:** Almacenamiento local del navegador donde se guarda el estado del diagrama que cuyo modelado esté en curso.
- **Navegador:** Medio a través del cual el usuario interactúa con la aplicación.

El siguiente diagrama ilustra estos componentes y su interacción con la aplicación:



Figura C.5: Diagrama con las partes de la aplicación en su ejecución en el navegador.

Apéndice D

Documentación técnica de programación

D.1. Introducción

Este anexo proporciona una guía detallada sobre la estructura de directorios del proyecto, un manual del programador, instrucciones para la compilación, instalación y ejecución del proyecto, así como una descripción de las pruebas del sistema. Está destinado a desarrolladores y posibles contribuidores que deseen comprender mejor el código y colaborar en su desarrollo.

D.2. Estructura de directorios

El proyecto está organizado de la siguiente manera:

```
.github/workflows
assets
docs
public
src/
  components/
    DiagramEditor/
      DiagramEditor.js
    utils/
  utils/
```

```
validation.js
sql.js
tests/
  e2e/
  unit/
  graphs/
```

- **.github/workflows:** Contiene archivos de configuración para las acciones de github y que establecen los procesos de integración y despliegue continuo.
- **assets:** Archivos de recursos para el proyecto.
- **docs:** Documentación del proyecto donde se encuentran los archivos LaTeX de esta memoria.
- **public:** Archivos que servirá el servidor web de la aplicación al levantarse.
- **src/components/DiagramEditor:** Contiene la inicialización de mxGraph, el canvas, la toolbar y toda la lógica para modelar el diagrama e interactuar con la aplicación. Hay una explicación un tanto más detallada en el Apéndice C.4.
- **src/utils:** Módulos para validar y exportar el diagrama como SQL.
- **tests/e2e:** Pruebas end-to-end.
- **tests/unit:** Pruebas unitarias.
- **tests/unit/graphs:** Diagramas JSON de ejemplo utilizados en las pruebas unitarias.

D.3. Manual del programador

Se proporciona una guía para configurar el proyecto y poder contribuir a su desarrollo.

Configuración del entorno de desarrollo

Para configurar el entorno de desarrollo, se necesitan las siguientes herramientas:

- Node.js (versión 18, el repositorio cuenta con un archivo `.node-version` que determina la versión node a usar.
- npm
- git

Clonación del proyecto

Uno de los primeros pasos será el de clonar el repositorio remoto a uno local. Preferiblemente se trabajará a través de un fork [5].

```
git clone git@github.com:rubenmate/draw-entity-relation.git
```

O si se estamos clonando a través del protocolo http (no recomendable, es mejor usar ssh).

```
git clone https://github.com/rubenmate/draw-entity-relation.git
```

Instalación de dependencias

Para instalar las dependencias del proyecto, ejecutar el siguiente comando en la raíz del proyecto:

```
npm install
```

Estructura del código

El código se organiza en módulos para facilitar su mantenimiento y comprensión. Los componentes principales de la aplicación se encuentran en `src/components/DiagramEditor`, que contiene toda la lógica para modelar diagramas E-R. Los módulos de validación y generación SQL se encuentran en `src/utils`.

Añadir nuevas funcionalidades

Para añadir nuevas funcionalidades, hay que seguir estos pasos:

1. Crear una rama nueva para su funcionalidad siguiendo la convención de nomenclatura:

```
git checkout -b <num. del issue>-funcionalidad
```

2. Realizar los cambios necesarios en el código.
3. Realizar los commits siguiendo la convención **conventional commits** [3].
4. Asegurarse de que las pruebas pasen correctamente.
5. Crear una pull request describiendo los cambios realizados.

Ejemplos de nombres de rama:

- Para una nueva funcionalidad: 29-diagram-persistence
- Para una corrección: 34-fix-toolbar-bug

Este proyecto cuenta con varias herramientas de integración continua:

Github Actions [6] que se ejecutan automáticamente al hacer commits a la rama principal main o en cualquier rama cuyo objetivo de integración sea la rama principal.

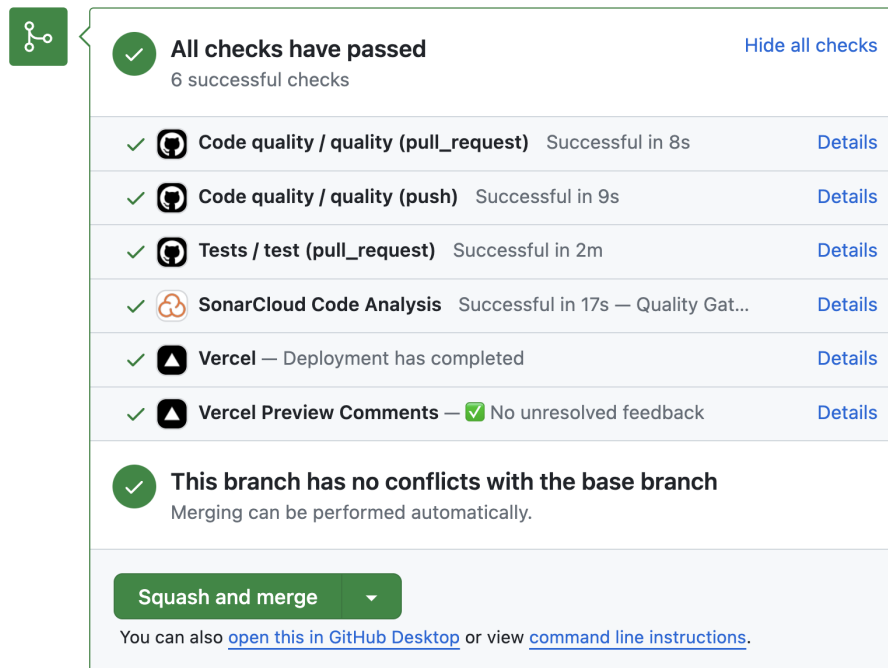


Figura D.1: Ejecución de las diferentes GitHub Actions en nuestro proyecto.

Las pruebas pueden ejecutarse también de forma local como se explica en el apartado de **Pruebas** posterior.

Al realizar un commit se ejecuta un hook que comprueba contra el código incluido en ese commit su corrección semántica según las reglas del linter Biome [2].

D.4. Compilación, instalación y ejecución del proyecto

Para compilar, instalar y ejecutar el proyecto, seguir estos pasos:

Compilación

Para compilar el proyecto, ejecutar el siguiente comando:

```
npm run build
```

Con esto se crearía la carpeta build, más información para desplegarla en la sección Despliegue.

Ejecución en modo de desarrollo

Para ejecutar el proyecto en modo de desarrollo, use el siguiente comando:

```
npm start
```

Se levantará el proyecto en la dirección por defecto:

```
http:localhost:3000
```

Aunque si dicho puerto estaría ocupado se preguntaría al usuario para levantarlo en otro.

Despliegue

Para desplegar el proyecto, se recomienda utilizar Vercel. La configuración es muy básica, tan solo es necesario iniciar sesión en su cuenta de Vercel y conectar con el repositorio del proyecto para que se realice el despliegue de manera automática (inicialmente y con cada nuevo commit).

En caso de querer realizar el despliegue en otra plataforma habría que hacer la build con el comando explicado en el apartado **Compilación**. En este punto ya se habría hecho la build y tan solo quedaría servirla con una utilidad como **serve** [9]

1. Instalar **serve** a nivel global en el sistema:

```
npm install -g serve
```

2. Hacer build del proyecto:

```
npm run build
```

3. Servir la aplicación con **serve**:

```
serve -s build
```

D.5. Pruebas del sistema

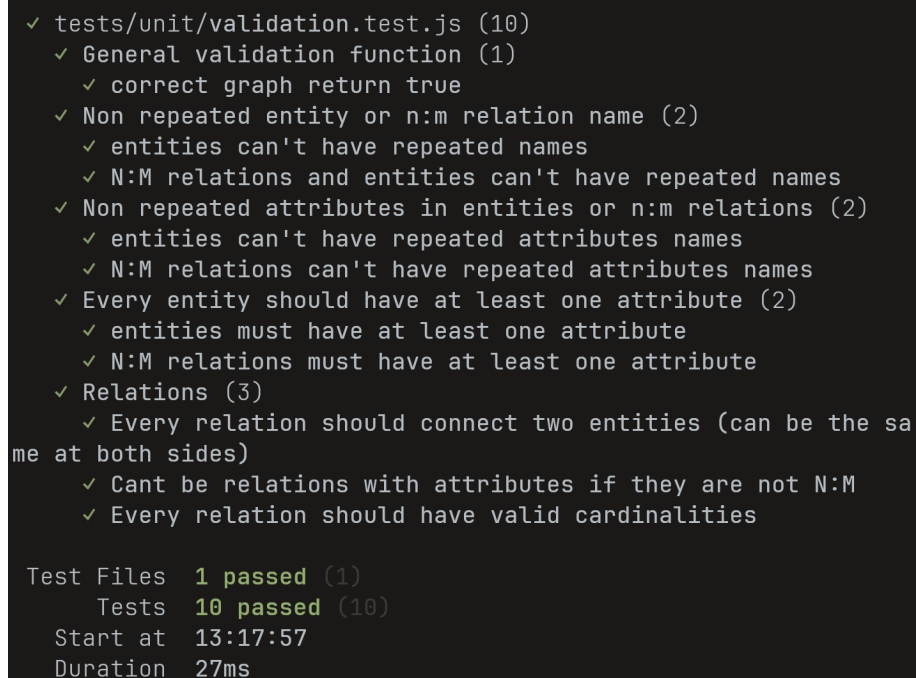
Las pruebas del sistema están divididas en pruebas end-to-end (e2e) y pruebas unitarias.

Pruebas unitarias

Las pruebas unitarias se encuentran en el directorio **tests/unit**. Estas pruebas verifican la funcionalidad de los componentes individuales del sistema. Los diagramas JSON de ejemplo utilizados en estas pruebas están en **tests/unit/graphs**.

Para ejecutar las pruebas unitarias, utilizar el siguiente comando:

```
npm run test
```



```
✓ tests/unit/validation.test.js (10)
  ✓ General validation function (1)
    ✓ correct graph return true
  ✓ Non repeated entity or n:m relation name (2)
    ✓ entities can't have repeated names
    ✓ N:M relations and entities can't have repeated names
  ✓ Non repeated attributes in entities or n:m relations (2)
    ✓ entities can't have repeated attributes names
    ✓ N:M relations can't have repeated attributes names
  ✓ Every entity should have at least one attribute (2)
    ✓ entities must have at least one attribute
    ✓ N:M relations must have at least one attribute
  ✓ Relations (3)
    ✓ Every relation should connect two entities (can be the same at both sides)
    ✓ Cant be relations with attributes if they are not N:M
    ✓ Every relation should have valid cardinalities

Test Files  1 passed (1)
Tests      10 passed (10)
Start at   13:17:57
Duration   27ms
```

Figura D.2: Ejecución de pruebas unitarias

Pruebas end-to-end

Las pruebas end-to-end se encuentran en el directorio **tests/e2e**. Estas pruebas verifican la funcionalidad completa del sistema desde un punto de vista de usuario final.

Para ejecutar las pruebas end-to-end, utilizar el siguiente comando:

```
npm run test:e2e
```

También se pueden lanzar de forma gráfica con

```
npx playwright test --ui
```

Con este comando se lanza la interfaz gráfica de la herramienta Playwright [12] desde donde podemos ver los diferentes tests y ejecutarlos gráficamente.

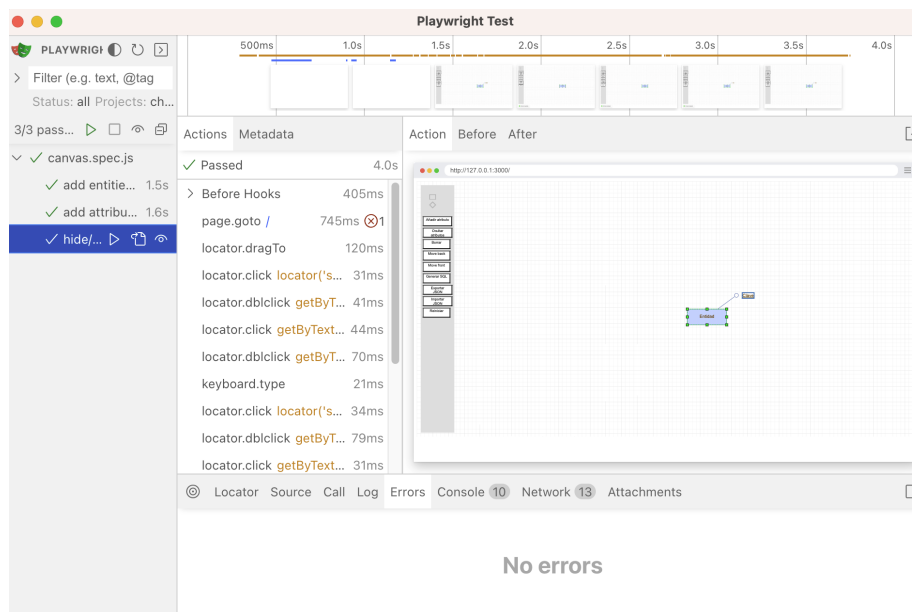


Figura D.3: Ejecución de pruebas End to End con la interfaz gráfica de Playwright

Apéndice *E*

Documentación de usuario

E.1. Introducción

En este apartado se proporciona una guía completa para los usuarios de la aplicación. Se detallan los requisitos necesarios, el proceso de instalación y un manual de usuario para facilitar el uso de la aplicación.

E.2. Requisitos de usuarios

Para utilizar esta aplicación, los usuarios deben cumplir con los siguientes requisitos:

- Disponer de un navegador web actualizado (Chrome, Firefox, Safari, Edge o similar).
- Conexión a internet.
- Es conveniente tener algún conocimiento del modelo E-R.

E.3. Instalación

La aplicación se puede utilizar directamente desde el navegador sin necesidad de instalación y es el modo recomendado de uso ¹. Sin embargo, para aquellos usuarios que deseen ejecutar la aplicación localmente, se pueden seguir estos pasos:

¹<https://draw-entity-relation.vercel.app>

1. Clonar el repositorio del proyecto desde GitHub:

```
git clone https://github.com/rubenmate/draw-entity-relation.git
```

2. Acceder al directorio del proyecto:

```
cd draw-entity-relation
```

3. Instalar las dependencias necesarias:

```
npm install
```

4. Iniciar el servidor de desarrollo:

```
npm start
```

5. Abrir navegador web y acceder a `http://localhost:3000`.

Para levantar la aplicación en un entorno de producción, se puede utilizar la herramienta `serve` de npm [9]:

1. Instalar `serve` a nivel global en el sistema:

```
npm install -g serve
```

2. Hacer build del proyecto:

```
npm run build
```

3. Servir la aplicación con `serve`:

```
serve -s build
```

E.4. Manual del usuario

Este manual proporciona una guía paso a paso sobre cómo utilizar las principales funcionalidades de la aplicación.

Inicio

Al entrar en la aplicación encontraremos las diferentes partes de la aplicación y el canvas vacío. Con el click derecho del ratón se puede hacer drag del canvas y movernos por si necesitásemos más espacio.



Figura E.1: Pantalla inicial de la aplicación.

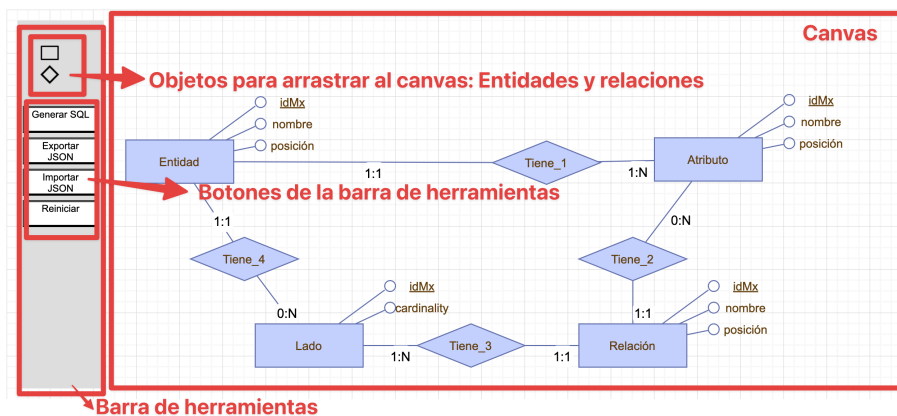


Figura E.2: Diferentes componentes de la aplicación.

Crear un nuevo diagrama E-R

1. En caso de tuviéramos un diagrama en proceso de modelar el proceso sería hacer click en el botón **Reiniciar** y **Aceptar** en el diálogo.
2. Con el canvas vacío el proceso de funcionamiento es tan simple como arrastrar los diferentes objetos (Entidades-rectángulos, Relaciones-Rombos) al canvas.

Acciones comunes sobre objetos

Con un objeto cualquiera (Entidades, Atributos y Relaciones) podemos realizar estas acciones:

1. Seleccionar un elemento: Al hacer click sobre un elemento se selecciona.
2. Deseleccionar un elemento: Al hacer click sobre un área vacía se deselecciona el elemento seleccionado.
3. Doble click para cambiar de nombre: Al hacer doble click sobre un objeto nos deja usar el teclado para cambiarlo de nombre.
4. Move back y Move front: Son comunes a todos los objetos y sirven para aquellos casos que queramos recuperar algún objeto que ha quedado detrás de otro. Con **Move back** el objeto queda por detrás del resto, **Move front** hace lo contrario.

Acciones sobre una Entidad

Con una entidad seleccionada podemos realizar las siguientes acciones:

1. Añadir atributo: Añade un atributo a esta entidad, el primer atributo siempre será clave (no así los siguientes) y la posición de los siguientes siempre será un poco debajo del último atributo añadido.
2. Ocultar atributos: Oculta los atributos de una entidad para evitar sobrecargar la visualización.
3. Mostrar atributos: Muestra los atributos de una entidad previamente ocultos.
4. Borrar: Elimina la entidad así como sus atributos y vinculación con relaciones si los hubiera.

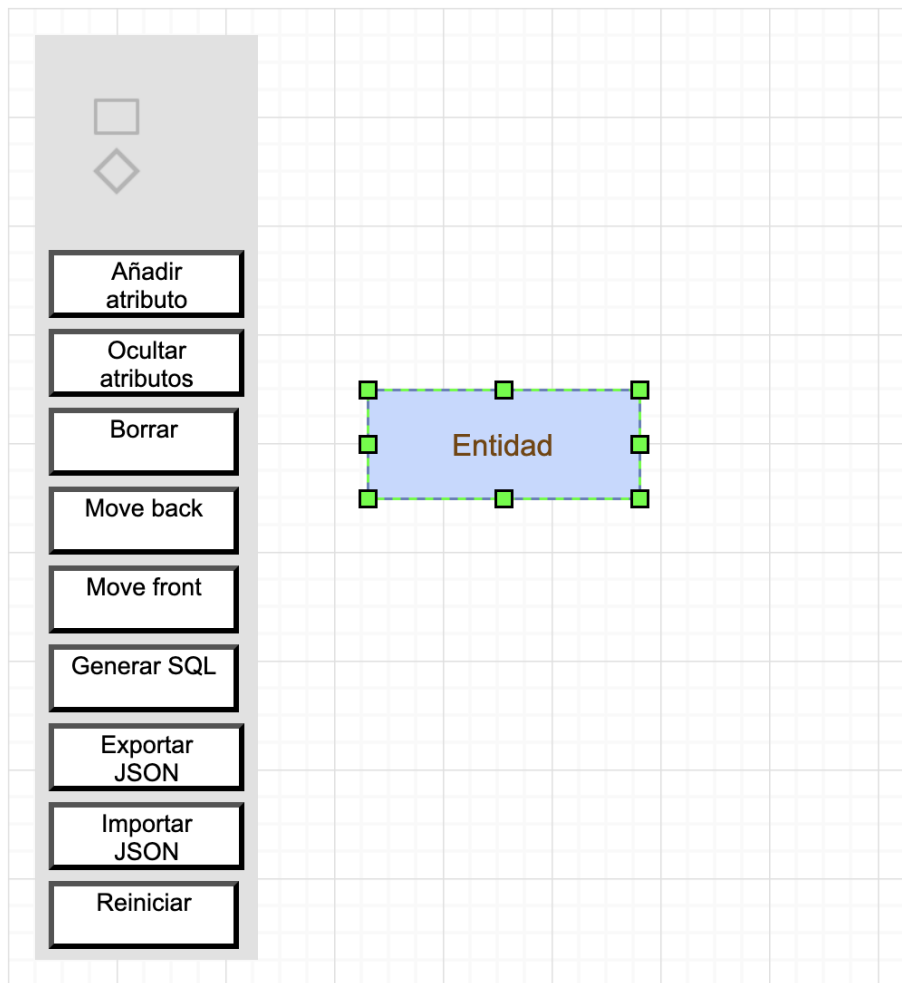


Figura E.3: Acciones contextuales sobre una entidad.

Acciones sobre un Atributo

Con un atributo seleccionado (no clave, el atributo clave está protegido) podemos realizar las siguientes acciones:

1. Convertir en clave: Convierte el atributo en clave. Como en esta versión no se permiten claves compuestas, el resultado será que la anterior clave pasará a ser atributo normal.
2. Borrar: Elimina el atributo de la entidad.

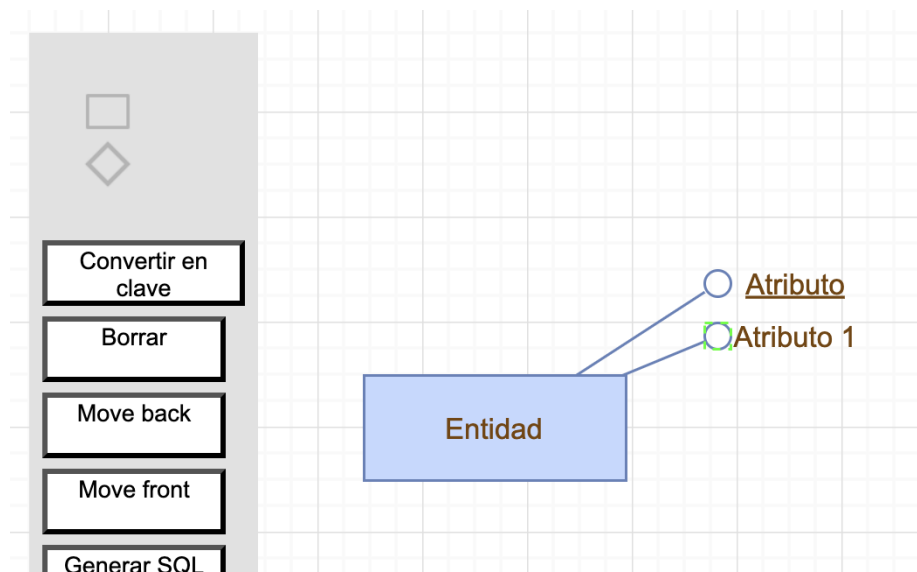


Figura E.4: Acciones contextuales sobre un atributo.

Acciones sobre una Relación

Con una relación seleccionada podemos realizar las siguientes acciones:

1. Convertir en clave: Convierte el atributo en clave. Como en esta versión no se permiten claves compuestas, el resultado será que la anterior clave pasará a ser atributo normal.
2. Borrar: Elimina el atributo de la entidad.

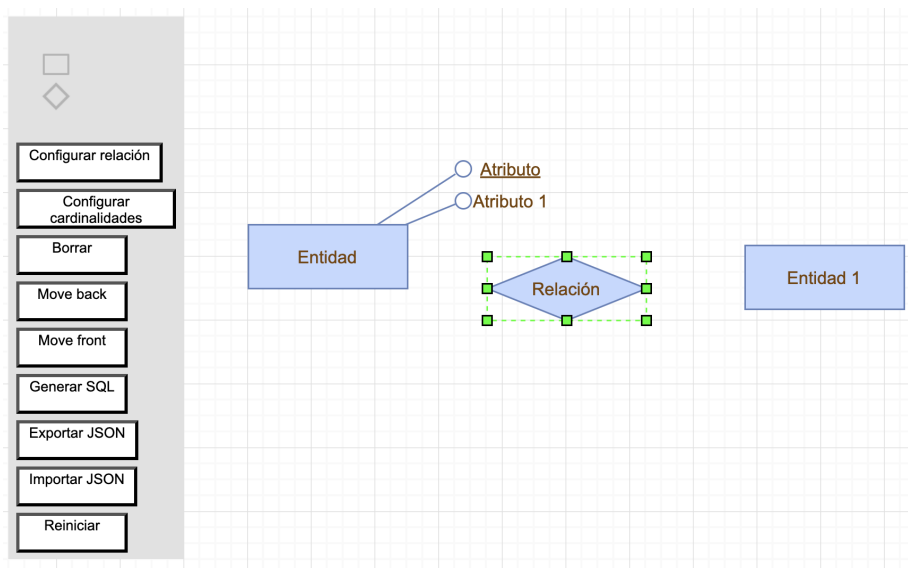


Figura E.5: Acciones contextuales sobre una relación.

Las relaciones N:M pueden contener atributos y por tanto también muestran los botones **Añadir atributo** o **Ocultar atributos** que se comportan del mismo modo que las entidades con la salvedad de que estos atributos no serán clave.

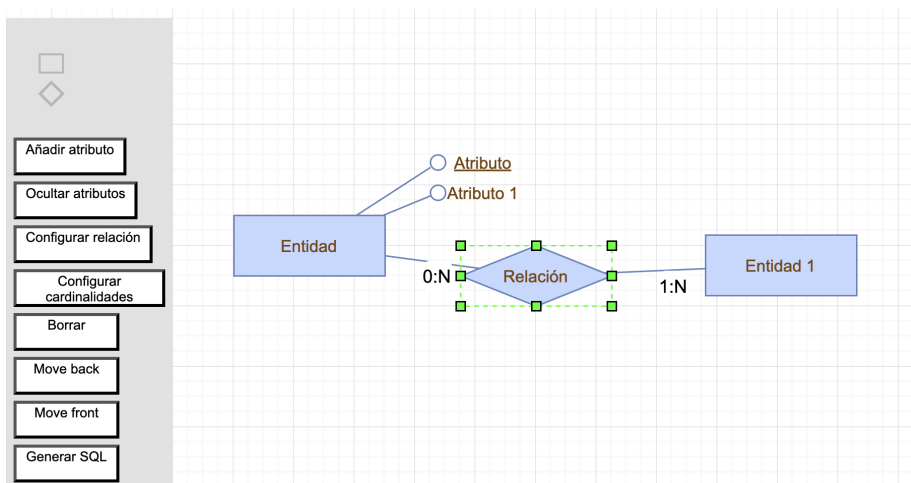


Figura E.6: Añadir atributos en una relación N-M.

Configurar relaciones

Para configurar una relación hay que hacer click sobre el botón **Configurar relación**, se mostrarán dos diálogos con selectores (que muestran las diferentes entidades del diagrama). Al escoger entidades en los dos lados (pueden ser la misma, se permiten relaciones reflexivas). Las relaciones se pueden reconfigurar con este mismo procedimiento.



Figura E.7: Diálogo para configurar relaciones.

Configurar cardinalidades

Para configurar las cardinalidades de los lados de una relación hay que hacer click sobre el botón **Configurar cardinalidades**. Se mostrarán dos diálogos con selectores (que muestran las posibles cardinalidades). El caso 1:1-1:1 no está permitido, al escoger 1:1 en uno de los dos lados

desaparecerá del otro. Las cardinalidades se pueden reconfigurar con este mismo procedimiento.

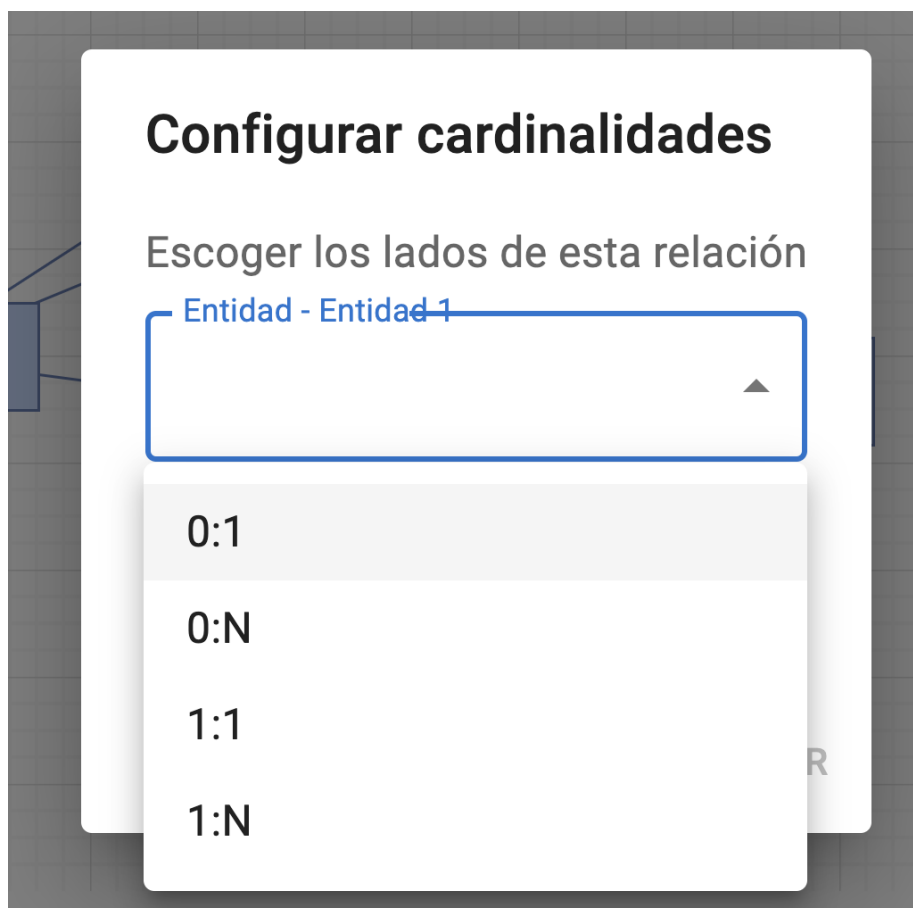


Figura E.8: Diálogo para configurar cardinalidades de una relación.

Generar script SQL

1. Al hacer click al botón **Generar SQL** se valida el diagrama internamente.
2. En caso de que el diagrama sea válido se generará un archivo SQL que se descargará automáticamente.
3. En caso de no ser válido se mostrará una serie de diagnósticos que están causando que el diagrama no sea válido.

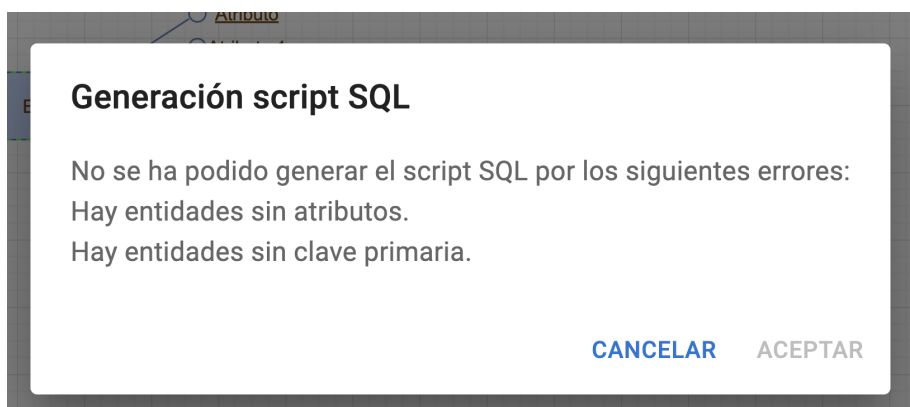


Figura E.9: Diagnóstico al generar SQL con un diagrama no válido.

Exportar un diagrama a JSON

El funcionamiento es exactamente igual que para generar un script SQL, pero con la salvedad de que se exporta el diagrama en formato JSON. Este archivo está listo para ser importado y recreado.

Importar un diagrama desde JSON

1. Haga clic en el botón **Importar JSON** y seleccione el archivo JSON que contiene el diagrama.
2. La aplicación comprobará la validez del diagrama y lo recreará.

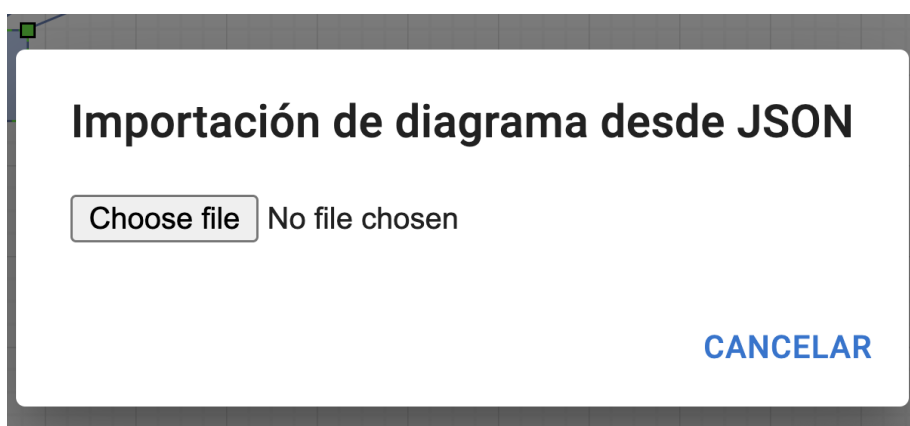


Figura E.10: Diálogo con selección de archivo para import diagrama JSON.

Reiniciar canvas

1. Haga clic en el botón **Reiniciar**.
2. Aparecerá un diálogo de confirmación con las opciones **Aceptar** y **Cancelar**.
3. Al hacer click en **Aceptar** se borrará el estado del diagrama y se reiniciará el canvas.

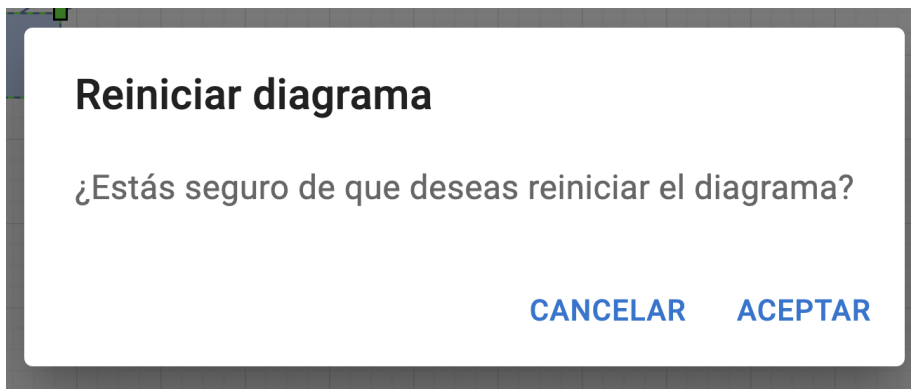


Figura E.11: Diálogo de confirmación para reiniciar el canvas.

Con estas instrucciones, los usuarios podrán utilizar la aplicación de manera efectiva para crear, validar, exportar e importar diagramas E-R, así como generar scripts SQL a partir de sus diagramas modelados.

Apéndice F

Anexo de sostenibilización curricular

F.1. Introducción

Este anexo incluye una reflexión personal sobre los aspectos de sostenibilidad abordados en este Trabajo de Fin de Grado (TFG). Se puede considerar que la sostenibilidad es una idea crucial en la formación académica actual y que además es fundamental para la creación de soluciones tecnológicas que respeten y promuevan el bienestar ambiental, social y económico.

A continuación, se detallan algunas de las competencias de sostenibilidad adquiridas y aplicadas durante el desarrollo del TFG, relacionándolas con los Objetivos de Desarrollo Sostenible (ODS) de la ONU.

F.2. Competencias de sostenibilidad adquiridas

ODS 4: Educación de Calidad

Durante el desarrollo de este TFG, se han adquirido y aplicado conocimientos avanzados en el ámbito del modelado de diagramas E-R y el desarrollo de aplicaciones web. Un desarrollo implementando buenas prácticas, tales como el uso de metodologías ágiles y herramientas de integración y despliegue continuos, contribuyen a una educación de calidad. Estos nuevos conocimientos no mejora simplemente mis habilidades y conocimientos téc-

nicos, sino que también estimulan un enfoque crítico y reflexivo al abordar la resolución de problemas complejos.

ODS 9: Industria, Innovación e Infraestructura

Este proyecto que se acaba de tratar se centra en la creación de una aplicación web para el modelado de diagramas E-R. Este tipo de herramientas son importantes para la educación, digitalización y modernización de infraestructuras educativas y empresariales. Al facilitar el aprendizaje, diseño y validación de bases de datos; esta aplicación promueve la eficiencia en esta tarea a la vez que se contribuye a la construcción de infraestructuras digitales resilientes y sostenibles.

ODS 12: Producción y Consumo Responsables

El desarrollo de este TFG ha incluido la utilización de recursos de manera eficiente y responsable. Se han usado librerías ya desarrolladas, como son React y mxGraph. Esto reduce la necesidad de recursos adicionales y promueve la reutilización de componentes existentes.

ODS 13: Acción por el Clima

Aunque no se puede decir que este TFG esté directamente relacionado con la mitigación del cambio climático, sí se puede decir que la adopción de prácticas sostenibles en el desarrollo de software tiene un impacto positivo en el medio ambiente. El uso de herramientas de integración continua y despliegue continuo reduce la necesidad de infraestructuras físicas y, como consecuencia, disminuye la huella de carbono asociada al desarrollo del software.

ODS 17: Alianzas para Lograr los Objetivos

El desarrollo del TFG ha implicado la colaboración con varias partes interesadas, como por ejemplo tutores académicos y otros desarrolladores con problemáticas comunes. Esta colaboración ha fomentado el intercambio de conocimientos, contribuyendo a la construcción de alianzas. El enfoque colaborativo y la comunicación son competencias clave que se han desarrollado y aplicado durante el proyecto, y que también son clave para el desarrollo sostenible.

F.3. Reflexión final

Como conclusión, el desarrollo de este TFG ha permitido la adquisición y aplicación de ciertas competencias de sostenibilidad que están alineadas con varios ODS (Objetivos de Desarrollo Sostenible). La implementación de buenas prácticas de desarrollo, la promoción de la innovación y la eficiencia, y el enfoque en la sostenibilidad ambiental y social son aspectos que han tenido presencia en el proyecto de una u otra manera. Esta experiencia ha mejorado mis habilidades técnicas. Pero no solo eso, también ha fortalecido mi compromiso con el desarrollo sostenible y responsable en el ámbito tecnológico.

Para más información sobre la sostenibilización curricular, se puede consultar el documento de la CRUE https://www.crue.org/wp-content/uploads/2020/02/Directrices_Sostenibilidad_Crue2012.pdf.

Bibliografía

- [1] Apache Software Foundation. Apache License, Version 2.0. <https://www.apache.org/licenses/LICENSE-2.0.html>. [Internet; descargado 7-julio-2024].
- [2] Biome. Biome. <https://biomejs.dev/>. [Internet; descargado 4-julio-2024].
- [3] Conventional Commits. Conventional commits 1.0.0. <https://www.conventionalcommits.org/en/v1.0.0/>. [Internet; descargado 7-julio-2024].
- [4] Facebook. <https://es.react.dev/>. <https://es.react.dev/>. [Internet; descargado 4-julio-2024].
- [5] GitHub. Fork a repo. <https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/working-with-forks/fork-a-repo>. [Internet; descargado 7-julio-2024].
- [6] GitHub. Github actions – documentation. <https://docs.github.com/en/actions>. [Internet; descargado 4-julio-2024].
- [7] jGraph. mxgraph. <https://github.com/jgraph/mxgraph>. [Internet; descargado 4-julio-2024].
- [8] maxGraph. maxgraph. <https://github.com/maxGraph/maxGraph>. [Internet; descargado 7-julio-2024].
- [9] npm. serve. <https://www.npmjs.com/package/serve>. [Internet; descargado 7-julio-2024].

- [10] Open Source Initiative. The MIT License. <https://mit-license.org/>. [Internet; descargado 7-julio-2024].
- [11] Sistemas Operativos. Licencia MIT. <https://sistemasoperativos.info/software-libre/licencia-mit/>. [Internet; descargado 7-julio-2024].
- [12] Playwright. Playwright. <https://playwright.dev/>. [Internet; descargado 4-julio-2024].
- [13] sistemasoperativos.info. Licencia Apache — Sistemas Operativos. <https://sistemasoperativos.info/software-libre/licencia-apache/>. [Internet; descargado 7-julio-2024].
- [14] Wikipedia. Javascript — wikipedia, la enciclopedia libre. <https://es.wikipedia.org/wiki/JavaScript>. [Internet; descargado 4-julio-2024].