Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática
Mención en Ingeniería de Software

Actualización y ampliación de funcionalidades de una web desarrollada con NodeJS

Autor:

D. Alejandro Pérez Maldonado

Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática
Mención en Ingeniería de Software

Actualización y ampliación de funcionalidades de una web desarrollada con NodeJS

Autor:

D. Alejandro Pérez Maldonado

Tutores:

Dr. Joaquín Adiego Rodríguez

Dra. Natalia Martín Cruz

Índice general

1 Introducción

1.1 Motivación

Este proyecto consiste en la actualización de una plataforma web desde la cual se pueden analizar datos relacionados con la eficiencia en la cooperación internacional para el desarrollo.

Esta plataforma ya fue actualizada a una tecnología más moderna por otro alumno en su Trabajo Fin de Grado. Por lo que en este caso la tarea a realizar consiste en ampliar funcionalidades que quedaron pendientes en su día o se han ido pensando que podrían ser útiles durante el uso de la plataforma y corregir algunos pequeños problemas que han ido surgiendo.

Mi principal motivación para escoger este trabajo fueron las tecnologías que usaba el mismo, las cuales están muy de moda a día de hoy y me parecería interesante aprender sobre ellas mientras desarrollaba el proyecto.

1.2 Objetivos del proyecto

Después de una reunión inicial, los principales objetivos que se han definido han sido:

- Corregir los errores encontrados en la web.
- Mejorar/ampliar el apartado de gráficos.
- Incluir la opción de consultar los datos de los proyectos junto a un nuevo rol de “visitante”.
- Poder exportar a Excel los datos de un proyecto.
- Cálculo automático de distintas medidas de eficiencia.
- Links a otras webs con información de cooperación para el desarrollo.
- Mejora de la interfaz y la página principal.
- Añadir un nuevo apartado “tablón de anuncios”.
- Poder importar proyectos a partir de los archivos Excel que la AECID permite descargar.

1.3 Estructura de la memoria

A continuación describiré la estructura seguida en la memoria.

- Introducción: En este primer punto se describirán los objetivos generales del proyecto de una forma resumida, estando dividido en los siguientes apartados:
    - Motivación
    - Objetivos del proyecto
    - Estructura de la memoria
    - Descripción del escenario
- Tecnologías utilizadas: Aquí se comenta, con algo de detalle las tecnologías que se utilizan en la aplicación y por lo tanto tendré que utilizar para el desarrollo del proyecto, constando de estos apartados:
    - MERN Stack
    - React
    - GraphQL
    - NodeJS
    - MongoDB
    - Npm
- Estructura actual de la web: Este apartado comentará la estructura actual de la aplicación, añadiendo los casos de uso y clases necesarios para las nuevas funcionalidades de esta:
    - Modelo de casos de uso
    - Análisis de requisitos
    - Diagrama de clases
    - Diagramas de actividad
    - Arquitectura
- Planteamiento del proyecto: En este punto comentaré el modelo de trabajo planificado y seguido en el desarrollo del proyecto, además de las tareas a realizar, planteadas como historias de usuario:
    - SCRUM
    - Planificación
    - Historias de usuario
    - Otras tareas a realizar
- Ejecución del proyecto: En esta parte comentaré como ha sido el desarrollo del proyecto, mostrando también los cambios en la interfaz derivados de los cambios realizados en la aplicación:

    - Estructura del proyecto
    - Desarrollo del proyecto
    - Cambios en la interfaz de la aplicación
- Conclusiones
- Bibliografía

1.4 Descripción del escenario

El objetivo general de este proyecto es la introducción de nuevas funcionalidades y la corrección de errores en la web https://caliope.infor.uva.es, una web en la que se gestionan proyectos sobre cooperación para el desarrollo.

Esta web fue creada inicialmente por Antonio Fuentes Pérez para la realización de su TFG. Unos años más tarde, fue actualizada a una segunda versión, ya más moderna por José Miguel Fernández Sáenz de Navarrete.

En este caso, se decidió utilizar como tecnologías principales ReactJS para el frontend, Node.JS para la parte del servidor y MongoDB para la base de datos.

Actualmente se ha detectado la necesidad de introducir mejoras tanto en la parte de gestión de los proyectos como en la parte de visualización de resultados. Junto a la corrección de algunos errores que se han ido descubriendo durante el uso de la web.

Los tutores que han orientado y revisado este trabajo son:

- **Joaquín Adiego Rodríguez**, doctor perteneciente al Departamento de Informática encargado de la Arquitectura y Tecnología de Computadores, Ciencias de la Computación e Inteligencia Artificial en la Escuela de Ingeniería Informática de Valladolid.
- **Natalia Martín Cruz**, profesora perteneciente al Departamento de Organización de Empresas y Comercialización e Investigación de Mercados en la Facultad de Ciencias Económicas y Empresariales de la Universidad de Valladolid.

Cómo la web ya estaba desarrollada y el trabajo consiste en una actualización de esta, las tecnologías que utilizaré para su desarrollo serán las citadas anteriormente y que explicaré con más detalle en el siguiente apartado.

2. Tecnologías utilizadas

2.1 MERN Stack

Un stack, o también llamado ecosistema de datos, es un conjunto de servicios utilizados para construir y ejecutar una única aplicación.

Entre los más utilizados podríamos encontrar MEAN, que utiliza MongoDB, Express.js, Angular y Node.js, o MERN, que será el que eligió utilizar en este caso el autor del TFG anterior y que por lo tanto seguiré utilizando en este. La única diferencia entre MERN y MEAN respecto a las tecnologías utilizadas es que el primero usa React en vez de Angular.

Este stack permite realizar fácilmente aplicaciones de tres niveles (frontend, backend y base de datos). React se encarga de la parte del frontend, para el backend utilizaremos Express.js, el cual es un framework de Node.js, y finalmente, para la base de datos será MongoDB la tecnología que se encargará de esta.

A continuación explicaré un poco más a fondo en qué consisten cada una de las tecnologías del stack.

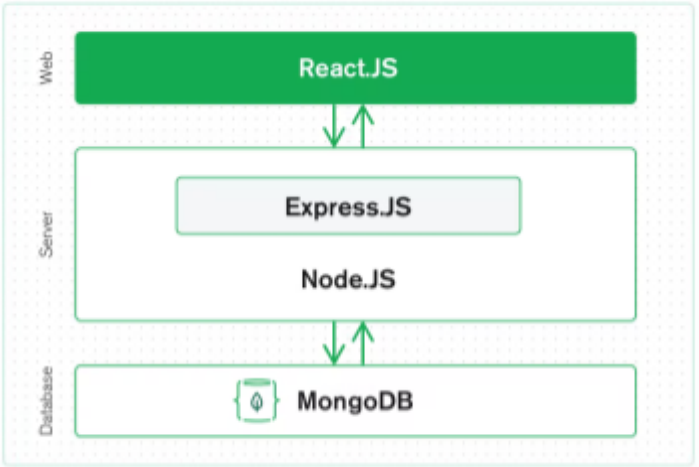

Figura 2.1: Stack MERN

2.2 React

React es una biblioteca de JavaScript focalizada en el desarrollo interfaces de usuario basada en componentes.

Esta fue desarrollada por Facebook inicialmente en el año 2011, ya que, detectaron que el típico marco de binding ralentizaba un poco su aplicación, debido a la cantidad de conexiones entre la vista y los datos. Fue utilizada por primera vez en el feed de noticias de la red social y , un año más tarde, en el año 2012, fue utilizada en Instagram, después de la adquisición de esta por parte de Facebook. Finalmente, en el año 2013, React fue liberada como código abierto, y a partir de entonces se ha ido convirtiendo en uno de los frameworks más utilizados para desarrollo web en la actualidad, utilizada por empresas como Netflix, Twitter o Paypal.

El elemento más importante de React es el componente, que, en esencia, es una pieza de la interfaz de usuario. Cuando desarrollamos una aplicación con React lo que estamos haciendo es crear componentes independientes y reusables para, poco a poco, ir creando interfaces de usuario más complejas.

Un componente básico estaría compuesto por unas “props”, un estado y un método “render”. Las “props” serían los valores que se le da al componente en el momento de su creación y estás nunca deben ser modificadas. Mientras que el estado contiene datos del componente que se pueden ir modificando a lo largo del ciclo de vida de este. Cualquier cambio en el estado del componente hará que se llame a la función “render”, que es la encargada de renderizar los elementos en el DOM.

React utiliza un DOM virtual. Su funcionamiento consiste en que, cada vez que se actualiza una vista, esta es renderizada en el DOM virtual, lo que es más rápido que actualizar el DOM real. Una vez hecho esto, compara ambos DOM para así saber cuales son las partes que es necesario actualizar en el navegador, y de esta forma conseguir una velocidad de carga más rápida.

2.2.1 Babel

JavaScript utiliza un estándar conocido como ECMAScript, o abreviado ES. Este estándar se va actualizando cada año añadiendo nuevas funcionalidades a JavaScript. Sin embargo, los navegadores no se actualizan al mismo ritmo que el estándar, por lo que, para poder utilizar estas últimas y novedosas funcionalidades necesitamos una herramienta que nos ayude a convertir el código.

Esta es la funcionalidad de Babel, es un “compilador” o transpilador de JavaScript que nos permite utilizar en nuestro código las últimas y novedosas

características de JavaScript y convertirlo en uno que sea entendido por navegadores que aún no han sido actualizados.

React utiliza JSX, una extensión de la sintaxis de JavaScript que nos permite insertar en el código etiquetas HTML directamente, facilitando de esta forma la legibilidad del código y la creación de los componentes.

Entonces Babel es el encargado de transformar este código JSX de React en código JavaScript normal que pueda ser interpretado correctamente por el navegador.

2.2.2 Webpack

Webpack es un empaquetador de módulos, dicho de otra forma, te permite generar un único archivo con todos aquellos módulos que necesita tu aplicación para funcionar.Es decir, te permite incluir todos los archivos javascript de la aplicación, incluso los archivos de estilos, en el mismo archivo, con nombre bundle.js. Además de esto también permite realizar otras tareas de optimización de los códigos, tales como la minificación y la compresión.

Webpack es una herramienta configurable que nos ayudará a realizar algunas tareas básicas en el desarrollo Frontend en tareas automatizadas y preparar nuestra aplicación web para producción.

Los siguientes son algunos conceptos básicos que nos ayudarán a entender cómo funciona la herramienta:

- **Entry point**: Indican los archivos que debe utilizar Webpack como entrada para generar los paquetes o archivos bundle.js.
- **Output**: Indican el lugar donde Webpack deberá colocar los paquetes bundle que se hayan generado: JavaScript, CSS, etc.
- **Loaders**: Son las rutinas que hacen posible que Webpack cargue, transforme y procese todos los archivos.
- **Plugins**: Amplían las funcionalidades que Webpack trae por defecto, permitiendo realizar tareas como la minificación o la ofuscación.

2.2.3 React router

React Router es una colección de componentes de navegación la cual podemos usar con React. Con esta librería vamos a obtener un enrutamiento dinámico gracias a los componentes, en otras palabras tenemos unas rutas que renderizan un componente.

Esta librería nos permite convertir nuestra aplicación React en una SPA (Single Page Application). Esta es un tipo de aplicación web que ejecuta todo su contenido en una sola página. Funciona cargando el contenido HTML, CSS y JavaScript por completo al abrir la web. Al ir pasando de una sección a otra, solo necesita cargar el contenido nuevo de forma dinámica si este lo requiere, pero no hace falta cargar la página por completo. Esto mejora los tiempos de respuesta y agiliza mucho la navegación, favoreciendo así a la experiencia de usuario.

Para evitar descargas excesivas de características inutilizadas, un SPA descarga progresivamente las características cuando se necesiten, pueden ser fragmentos de las páginas o módulos completos de la pantalla.

2.3 GraphQL

GraphQL fue creada en Facebook porque la empresa necesitaba resolver muchos problemas técnicos con su aplicación móvil nativa, de manera que GraphQL permite mejorar la comunicación entre las diferentes partes de una aplicación de software, por lo que hace que esa aplicación sea más fácil de entender, desarrollar, mantener y escalar.

GraphQL engloba dos elementos principalmente, por un lado un lenguaje de consulta que le permite a los clientes que consumen un servicio web, especificar qué datos necesitan. Por otro lado, es un entorno de ejecución para responder a estas consultas a través de la especificación de un esquema tipado en el que se enlistan los datos que el servicio web puede entregar y las operaciones para dar respuesta a las solicitudes de los clientes.

A diferencia de las alternativas a GraphQL donde las respuestas para una consulta están predefinidas y no pueden modificarse por parte del cliente, las consultas de GraphQL son dinámicas, el cliente dice qué quiere y en qué formato lo quiere. Por lo que minimiza la cantidad de datos que deben transferirse a través de la red y, por lo tanto, mejora considerablemente las aplicaciones que operan en circunstancias por ejemplo en donde la red no sea eficiente o el dispositivo sea de baja potencia.

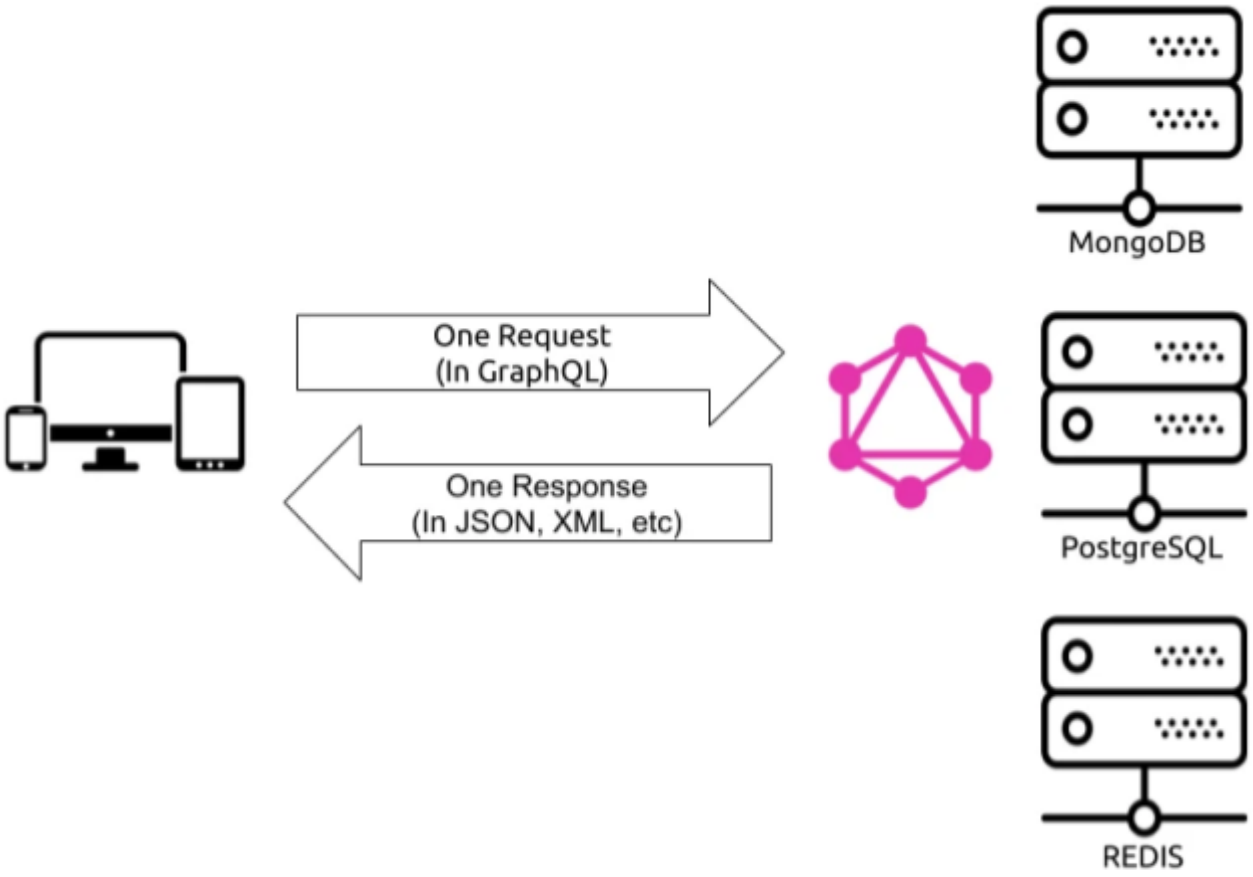

Figura 2.2: Comunicación de GraphQL

2.4 NodeJS

Node.js es un entorno de tiempo de ejecución de JavaScript. Este entorno de tiempo de ejecución en tiempo real incluye todo lo que se necesita para ejecutar un programa escrito en JavaScript. También aporta muchos beneficios y soluciona muchísimos problemas.

Node.js fue creado por los desarrolladores originales de JavaScript. Lo transformaron de algo que solo podía ejecutarse en el navegador en algo que se podría ejecutar en los ordenadores como si de aplicaciones independientes se tratara. Gracias a Node.js se puede ir un paso más allá en la programación con JavaScript no solo creando sitios web interactivos, sino teniendo la capacidad de hacer cosas que otros lenguajes de secuencia de comandos como Python pueden crear.

Tanto JavaScript como Node.js se ejecutan en el motor de tiempo de ejecución JavaScript V8 (V8 es el nombre del motor de JavaScript que alimenta

Google Chrome. Es lo que toma nuestro JavaScript y lo ejecuta mientras navega con Chrome). Este motor coge el código JavaScript y lo convierte en un código de máquina más rápido. El código de máquina es un código de nivel más bajo que la computadora puede ejecutar sin necesidad de interpretarlo primero, ignorando la compilación y por lo tanto aumentando su velocidad.

Node.js utiliza un modelo de entrada y salida sin bloqueo controlado por eventos que lo hace ligero y eficiente. Puede referirse a cualquier operación, desde leer o escribir archivos de cualquier tipo hasta hacer una solicitud HTTP.

Donde Node.js realmente brilla es en la creación de aplicaciones de red rápidas, ya que es capaz de manejar una gran cantidad de conexiones simultáneas con un alto nivel de rendimiento, lo que equivale a una alta escalabilidad.

2.4.1 ExpressJS

Express es el framework web más popular de Node, y es la librería subyacente para un gran número de otros frameworks web de Node populares. Proporciona mecanismos para:

- Escritura de manejadores de peticiones con diferentes verbos HTTP en diferentes rutas.
- Integración con motores de renderización de "vistas" para generar respuestas mediante la introducción de datos en plantillas.
- Establecer ajustes de aplicaciones web como qué puerto usar para conectar, y la localización de las plantillas que se utilizan para renderizar la respuesta.
- Añadir procesamiento de peticiones "middleware" adicional en cualquier punto dentro de la tubería de manejo de la petición.

A pesar de que Express es en sí mismo bastante minimalista, los desarrolladores han creado paquetes de middleware compatibles para abordar casi cualquier problema de desarrollo web. Hay librerías para trabajar con cookies, sesiones, inicios de sesión de usuario, parámetros URL, datos POST, cabeceras de seguridad y muchos más.

2.5 MongoDB

MongoDB es un sistema de base de datos NoSQL orientado a documentos de código abierto y escrito en C++. Al ser un proyecto de código abierto, sus binarios están disponibles para los sistemas operativos Windows, GNU/Linux, OS X y Solaris y es usado en múltiples proyectos o implementaciones en empresas como MTV Network, Craigslist, BCI o Foursquare. Algunas de sus características son:

- MongoDB almacena datos en documentos flexibles similares a JSON, por lo que los campos pueden variar entre documentos y la estructura de datos puede cambiarse con el tiempo.

- El modelo de documento se asigna a los objetos en el código de su aplicación para facilitar el trabajo con los datos.

- Las consultas ad hoc, la indexación y la agregación en tiempo real ofrecen maneras potentes de acceder a los datos y analizarlos.

- MongoDB es una base de datos distribuida en su núcleo, por lo que la alta disponibilidad, la escalabilidad horizontal y la distribución geográfica están integradas y son fáciles de usar.

Cualquier aplicación que necesite almacenar datos semi estructurados puede usar MongoDB. Es el caso de las típicas aplicaciones CRUD o de muchos de los desarrollos web actuales.

Eso sí, aunque las colecciones de MongoDB no necesitan definir un esquema, es importante que diseñemos nuestra aplicación para seguir uno. Tendremos que pensar si necesitamos normalizar los datos, denormalizarlos o utilizar una aproximación híbrida. Estas decisiones pueden afectar al rendimiento de nuestra aplicación. En definitiva el esquema lo definen las consultas que vayamos a realizar con más frecuencia.

MongoDB es especialmente útil en entornos que requieran escalabilidad. Con sus opciones de replicación y sharding, que son muy sencillas de configurar, podemos conseguir un sistema que escale horizontalmente sin demasiados problemas.

2.5.1 Mongoose

Mongoose es una librería para Node.js que nos permite escribir consultas para una base de datos de MongoDB, con características como validaciones, construcción de queries, middlewares, conversión de tipos y algunas otras, que enriquecen la funcionalidad de la base de datos.

La parte central del uso de Mongoose está en la definición de un esquema donde se indica la configuración de los documentos para una colección de MongoDB. Y aunque MongoDB es una base de datos NoSQL, donde los documentos se almacenan sin un esquema predefinido, el uso de un esquema te permite normalizar tu información, sin sacrificar la flexibilidad. Además, hace que la transición de SQL a NoSQL, sea más sencilla.

En el esquema mencionado especificamos los campos que pertenecen a un documento, validaciones y configuraciones especiales para su consulta. El esquema es, además, el lugar que nos permite enriquecer la funcionalidad de nuestros documentos de mongoose, ya sea vía la definición de campos virtuales, middlewares, métodos especiales para los objetos, entre otros.

En palabras prácticas, Mongoose funciona como una capa adicional sobre MongoDB a través de la cuál se implementan y automatizan muchas de las tareas habituales de trabajar con una base de datos.

Mongoose además, abre las puertas a una comunidad de plugins que puedes usar para automatizar tareas comunes, tales como el encriptado de información, paginación, consultas adicionales, y más.

2.6 Npm

npm, Node Package Manager, es un gestor de paquetes para JavaScript. Su objetivo es llevar el “desarrollo en JavaScript a la elegancia, productividad, y seguridad” y que no tengas que reinventar la rueda si quieres resolver problemas que la comunidad ya ha dado solución y otros proyectos con las mismas necesidades.

Este está organizado en tres partes o componentes:

- El portal, un sitio web que te permite encontrar paquetes de terceros, gestionar tus paquetes o configurar perfiles.

- La interfaz de línea de comandos de npm que te permite interactuar, instalar y publicar paquetes.

- El Registro, es una colección pública de paquetes de código open-source para la mayoría de las necesidades de la comunidad JavaScript.

El fichero package.json contiene información importante que npm usa para identificar el proyecto y gestionar las dependencias. Cuando instalas un paquete y este tiene dependencias, npm instalará también las dependencias de las dependencias.

3 Estructura actual y análisis de la web

3.1 Modelo de casos de uso

En la siguiente figura se muestra el diagrama de casos de uso de la web antes de la realización de la actualización:

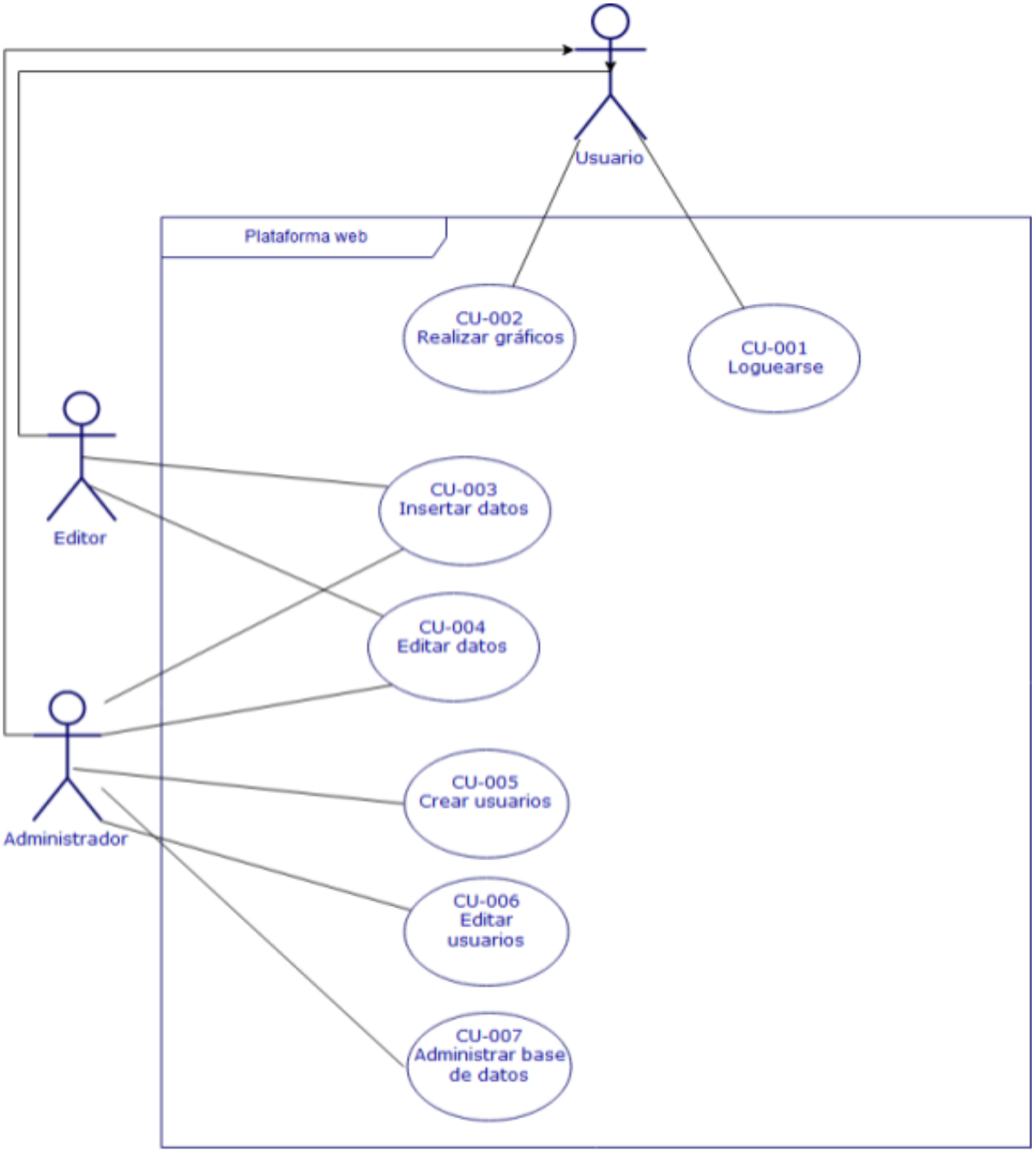

Figura 3.1: Diagrama de los casos de uso. Fernández [2019]

Y a continuación se muestra como quedaría el nuevo diagrama añadiendo los nuevos casos de uso y el nuevo rol.

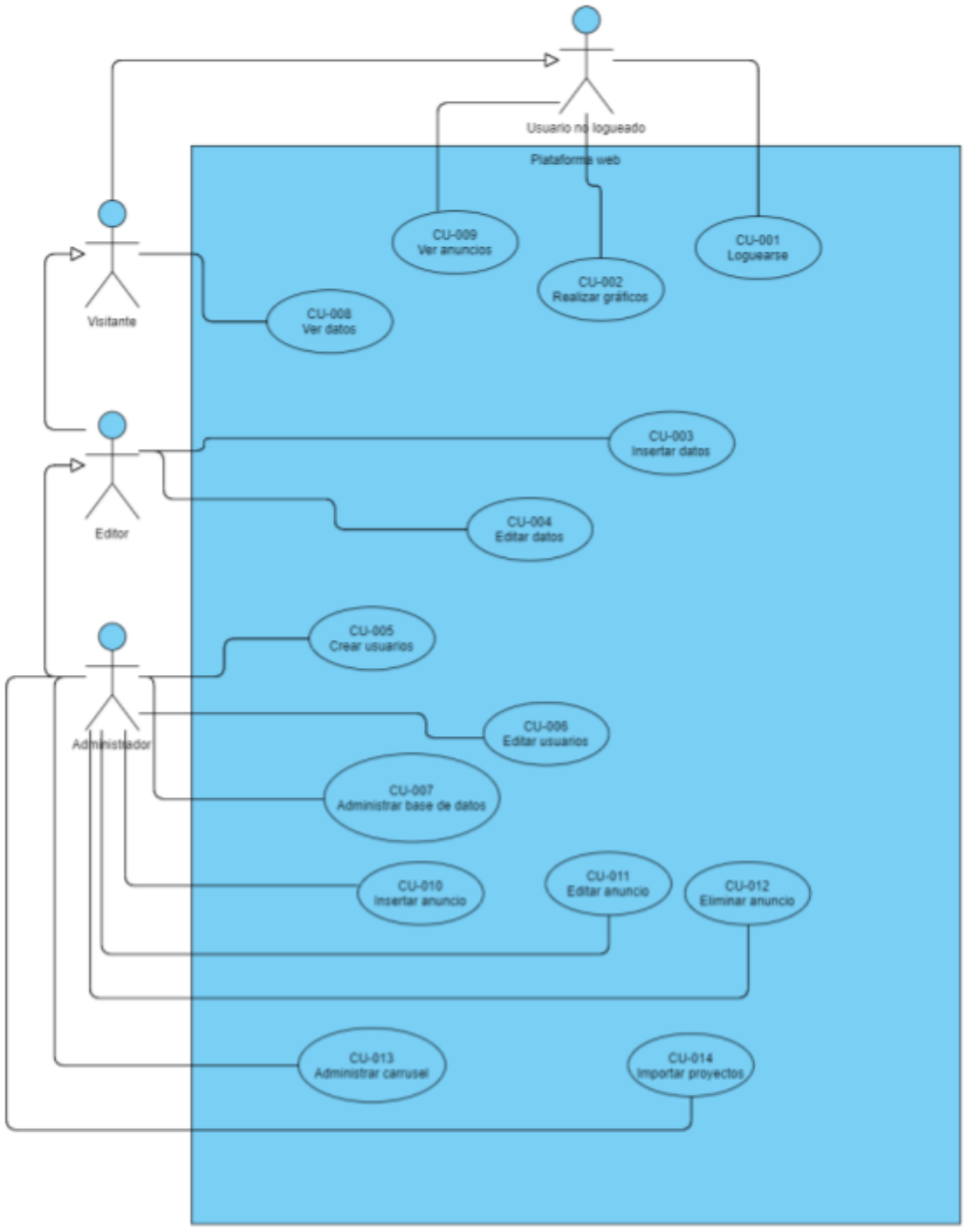

Figura 3.2: Diagrama de los casos de uso actualizado

Como se puede observar se ha añadido un nuevo actor, el cual es un usuario logueado en la web, con el rol de visitante, lo que le permite simplemente poder ver los datos de los proyectos.

El resto de casos de uso que se han añadido son:

- Para el actor “Usuario sin loguear”
    - Ver anuncios: Para ver anuncios en la nueva sección que se ha añadido, en la que se irán publicando avisos relacionados con la web y contenidos relacionados, no será necesario estar logueado.
- Para el actor “Administrador”
    - Todo lo relacionado con añadir, editar y eliminar anuncios de la web.
    - Administrar carrusel: Se ha sustituido la imagen de la parte inferior de la página principal por un carrusel de imágenes, al que podrá añadir o eliminar las imágenes que quiera que se muestren en él.
    - Importar proyectos: Ahora el usuario con rol de Administrador puede importar proyectos en formato Excel que proporciona la AECID.

3.2 Análisis de requisitos

En este apartado se enumerarán los requisitos anteriores del sistema, actualizados si es necesario, y los nuevos añadidos con las nuevas funcionalidades de la aplicación.

3.2.1 Requisitos funcionales

Identificación	*RF-01*
Descripción	*El sistema deberá permitir la inserción de nuevos proyectos y datos asociados a ellos.*

Identificación	*RF-02*
Descripción	*El sistema deberá permitir la edición de los proyectos y datos asociados a ellos que ya se encuentren registrados en el sistema.*

Identificación	*RF-03*
Descripción	*El sistema deberá permitir la visualización de gráficas extraídas de los datos seleccionados por el usuario.*

Identificación	*RF-04*
Descripción	*El sistema deberá proporcionar una interfaz gráfica a través de la cual el usuario puede acceder a las diferentes funcionalidades de la plataforma.*

Identificación	*RF-05*
Descripción	*El sistema organizará el acceso a la información a través de la interfaz dividiendo los apartados en:* ● *Datos: apartado en el cual se insertarán, editarán y visualizarán los datos referentes a los proyectos.* ● *Gráficas: apartado en el cual se mostrarán las gráficas extraídas de los datos.* ● *Dirección de proyectos: apartado desde el que se gestionarán y visualizarán los anuncios internos de la página.* ● *Tablón de anuncios: apartado desde el que se gestionarán y visualizarán los anuncios públicos de la web.* ● *Usuarios: apartado para la gestión de usuarios del sistema.* ● *Carrusel: apartado desde el que se gestionan las fotos que aparecen en la página de inicio.* ● *Importar AECID: apartado en el que se podrá subir un archivo Excel de la AECID para importar sus proyectos a la base de datos de la web.*

Identificación	*RF-06*
Descripción	*El sistema controlará la inserción de los datos a través de formularios, asegurándose que los datos introducidos son válidos.*

Identificación	*RF-07*
Descripción	*El sistema deberá tener un sistema de acceso a la base de datos en la cual podrán visualizarse todos los datos guardados en la plataforma.*

Identificación	*RF-08*
Descripción	*El sistema poseerá los siguientes roles de usuario:* ● *Administrador: Tendrá acceso a todos los apartados del sistema.* ● *Editor: Tendrá acceso a todos los apartados excepto a la gestión de usuarios, la gestión del carrusel, la gestión de anuncios internos y la importación de proyectos de la* AECID. ● *Visitante: Tendrá acceso a los mismos apartados que el Editor, pero solo podrá visualizar los proyectos y no podrá gestionar anuncios públicos.* ● *Usuario no registrado: Solamente podrá visualizar los apartado de Gráficas y Tablón de anuncios.*

Identificación	*RF-09*
Descripción	*El sistema llevará un registro de los usuarios registrados y permitirá registrar a nuevos usuarios a través de un apartado designado para ello al que solo podrán acceder los administradores.*

Identificación	*RF-10*
Descripción	*El sistema deberá permitir la visualización, sin posibilidad de edición, de los proyectos y datos asociados a ellos que ya se encuentren registrados en el sistema.*

Identificación	*RF-11*
Descripción	*El sistema deberá permitir filtrar los proyectos por identificador, país, año o título.*

Identificación	*RF-12*
Descripción	*El sistema deberá permitir escribir en un cuadro de texto para filtrar el listado de proyectos a seleccionar.*

Identificación	*RF-13*
Descripción	*El sistema deberá permitir exportar a un archivo Excel los datos de un proyecto.*

Identificación	*RF-14*
Descripción	*El sistema deberá permitir seleccionar diferentes tipos de gráfica para visualizar.*

Identificación	*RF-15*
Descripción	*El sistema deberá permitir la creación de anuncios, públicos o privados.*

Identificación	*RF-16*
Descripción	*El sistema deberá permitir la edición de anuncios, públicos o privados, incluyendo la posibilidad de cambiar la privacidad del mismo.*

Identificación	*RF-17*
Descripción	*El sistema deberá permitir la gestión de imágenes del carrusel de imágenes de la página de inicio.*

Identificación	*RF-18*
Descripción	*El sistema deberá permitir la importación de los proyectos incluídos dentro de un archivo Excel con el formato que proporciona la* AECID.

3.2.2 Requisitos no funcionales

Identificación	*RNF-01*
Descripción	*El sistema deberá estar instalado en un servidor de la universidad, al tratarse de un sistema multiplataforma, el sistema operativo queda a elección del cliente.*

Identificación	*RNF-02*
Descripción	*El sistema deberá ser accesible de forma pública desde Internet, aunque existan apartados que estén cerrados a los usuarios que no estén registrados en la plataforma.*

Identificación	*RNF-03*
Descripción	*El sistema deberá estar adaptado para poderse acceder desde plataformas de escritorio y móviles.*

Identificación	*RNF-04*
Descripción	*El sistema no deberá exceder unos tiempos máximos de carga de 3 segundos para todas las interfaces y de 5 para la generación de gráficas.*

Identificación	*RNF-05*
Descripción	*El sistema deberá solicitar una confirmación para cualquier borrado de datos que se produzca en la base de datos.*

Identificación	*RNF-06*
Descripción	*El sistema deberá mostrar una "rueda de carga" para los procesos que puedan tardar más de 3 segundos en visualizarse.*

3.3 Diagrama de clases

En la siguiente figura 2.3 se muestra el diagrama de clases de la web antes de la realización de la actualización:

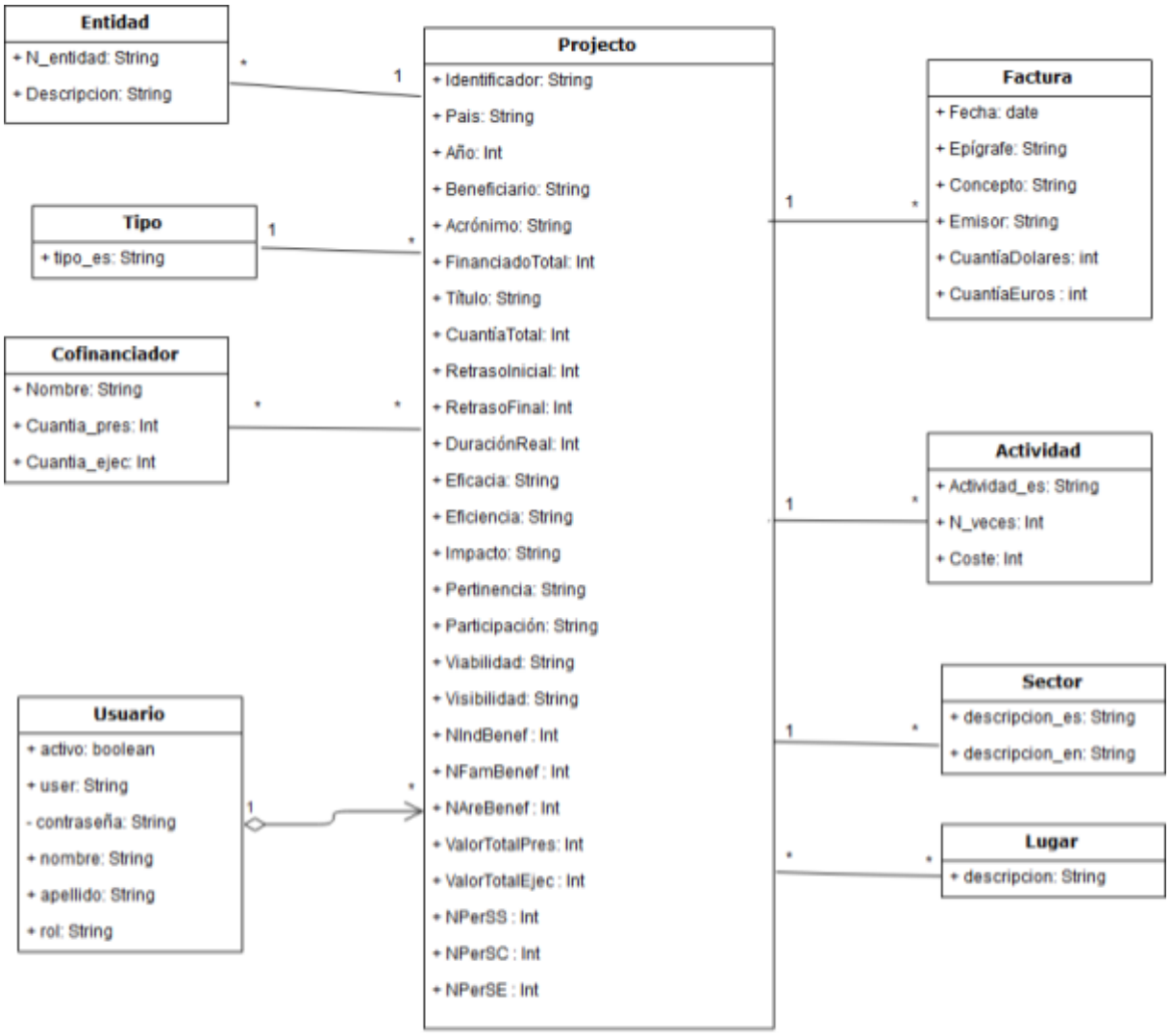

Figura 3.3: Diagrama de clases. Fernández [2019]

Para el diagrama de clases, simplemente se añadirán dos nuevas clases. Una para los anuncios de los nuevos apartados “Tablón de anuncios” y “Dirección de Proyectos”, con los atributos Título, Descripción, Imagen, que contendrá la ruta en la que se encuentra almacenada la imagen del anuncio, Archivo, lo mismo que el anterior pero guardará la ruta del archivo adjunto de la noticia y Público, que indicará en cuál de los apartados se mostrará el anuncio en función de si es verdadero o falso. Y la segunda será la clase Imagen, en la que se almacenará la

ruta de cada imagen subida al carrusel de imágenes añadido en el apartado inicial de la web.

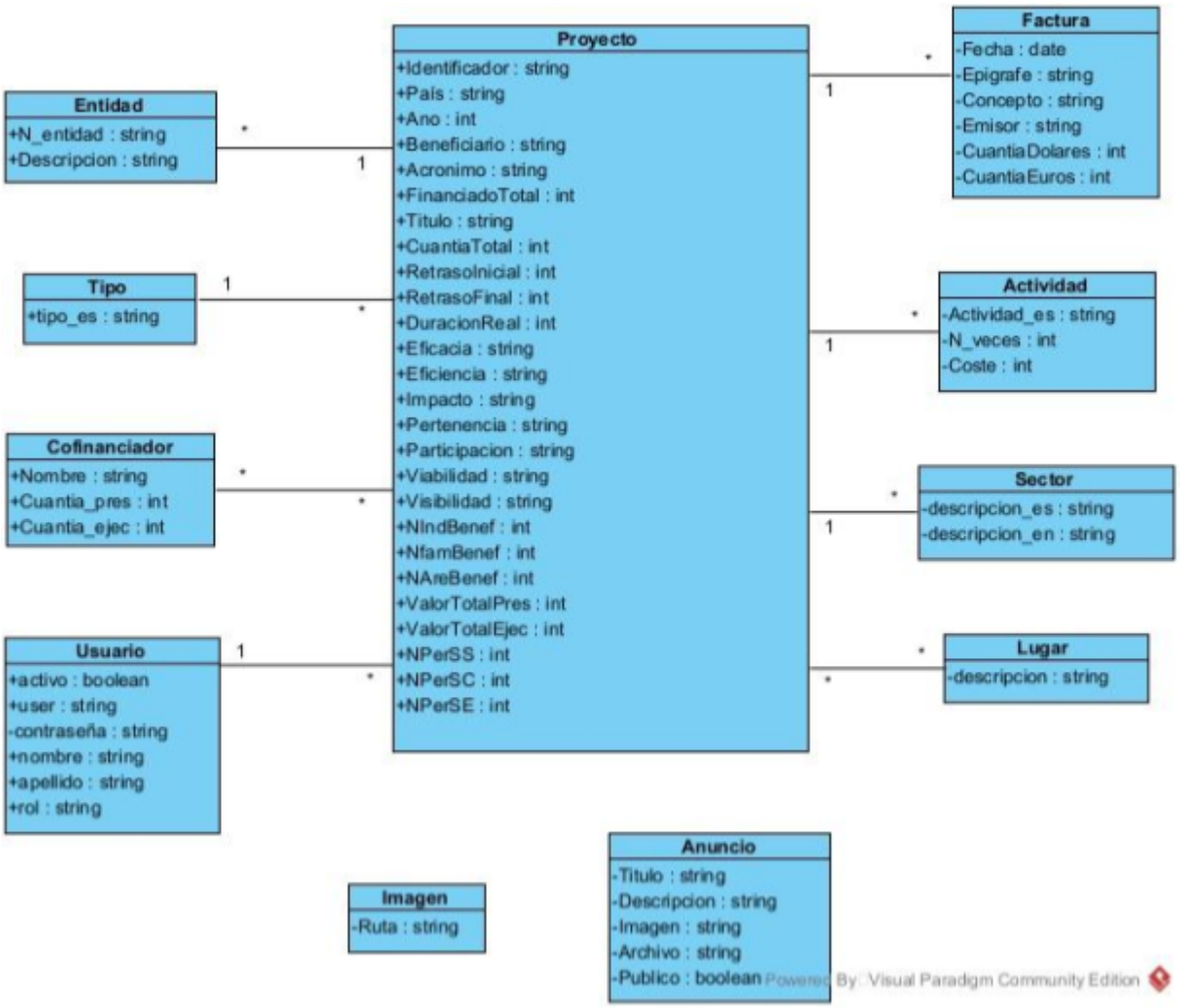

Figura 3.4: Diagrama de clases actualizado

3.4 Diagramas de actividad

En este apartado se detallarán los diagramas de actividad de varios de los principales procesos de la web.

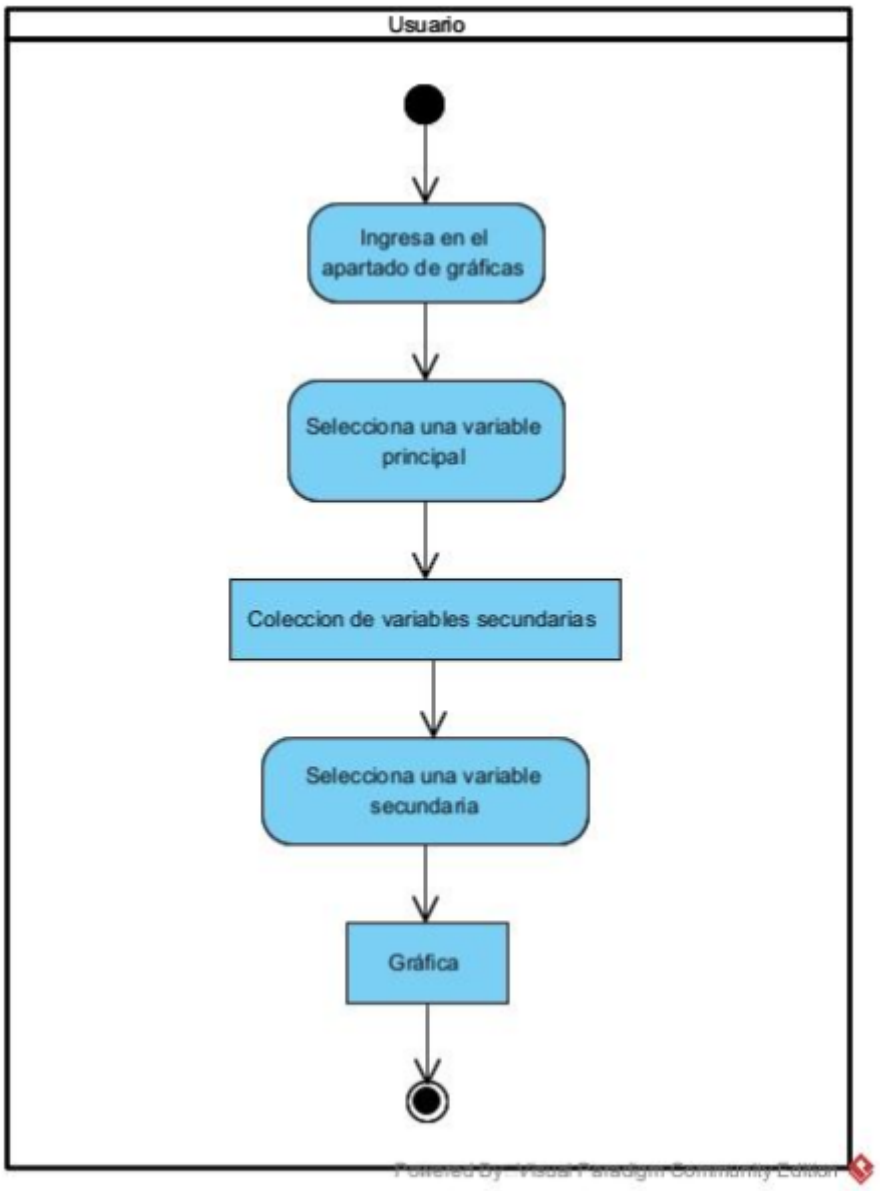

Figura 3.5: Diagrama de actividad del proceso de generación una gráfica.

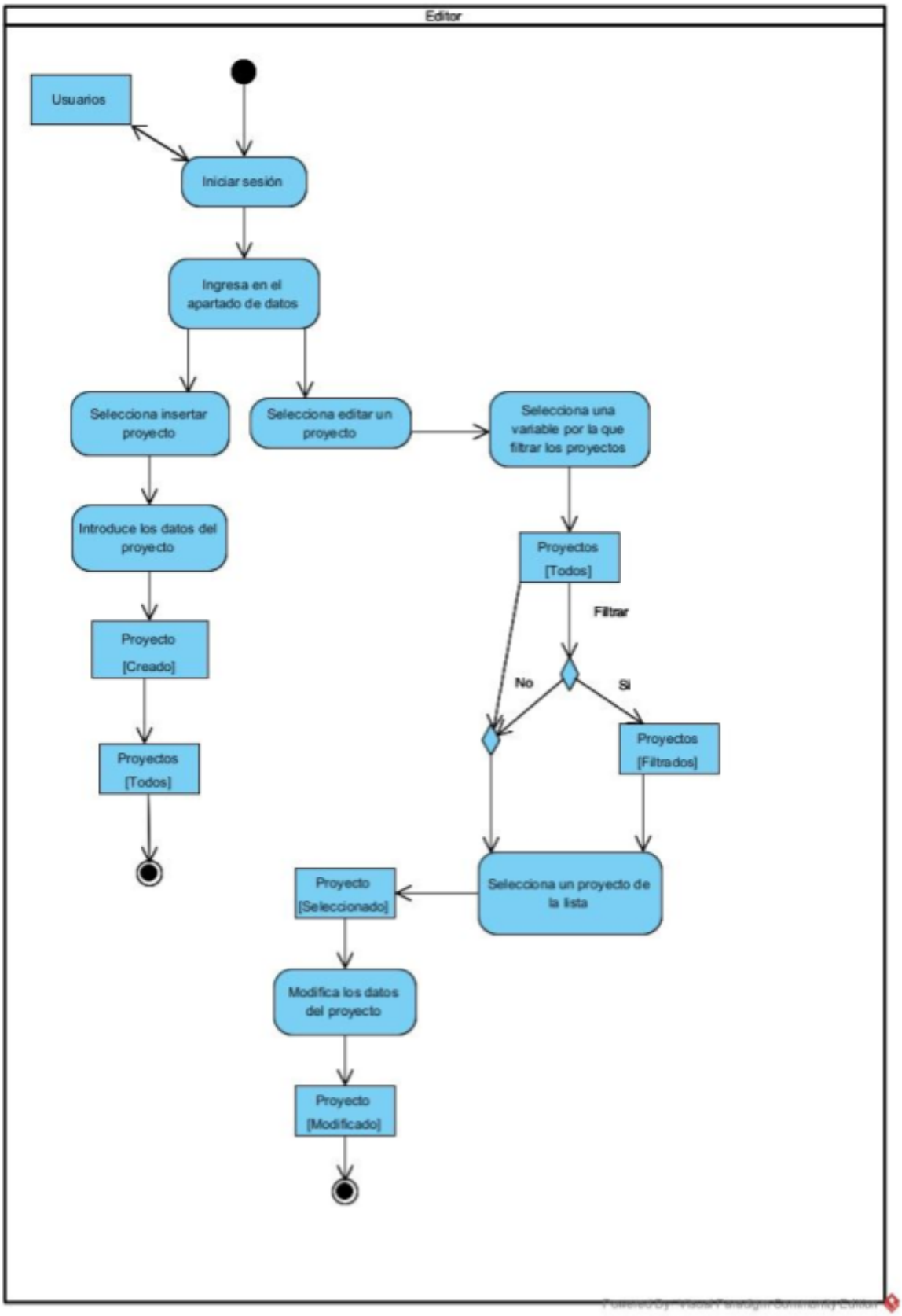

Figura 3.6: Diagrama de actividad de los procesos de creación y edición de un proyecto

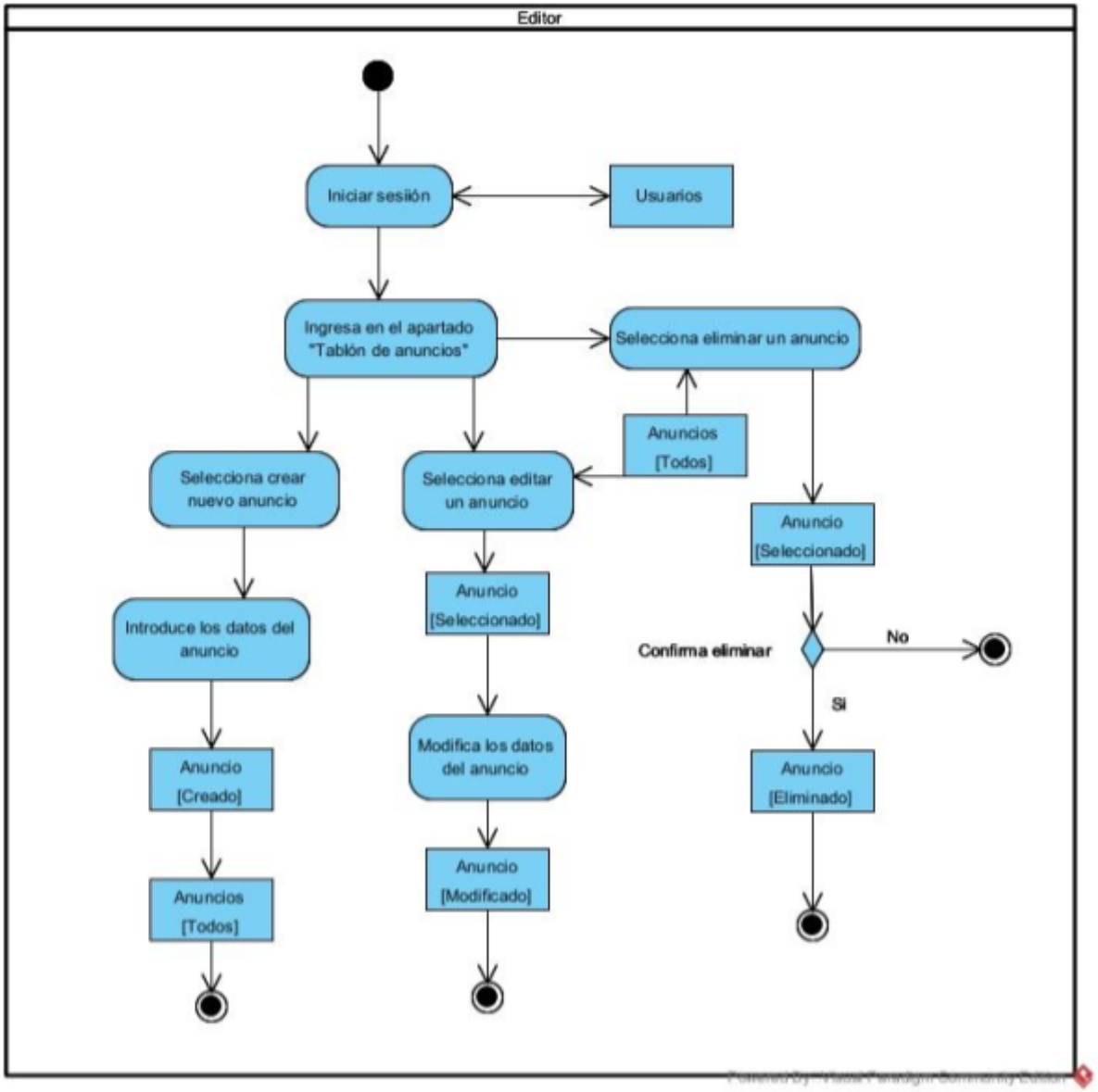

Figura 3.7: Diagrama de actividad de los procesos de creación, edición, y eliminación de un anuncio.

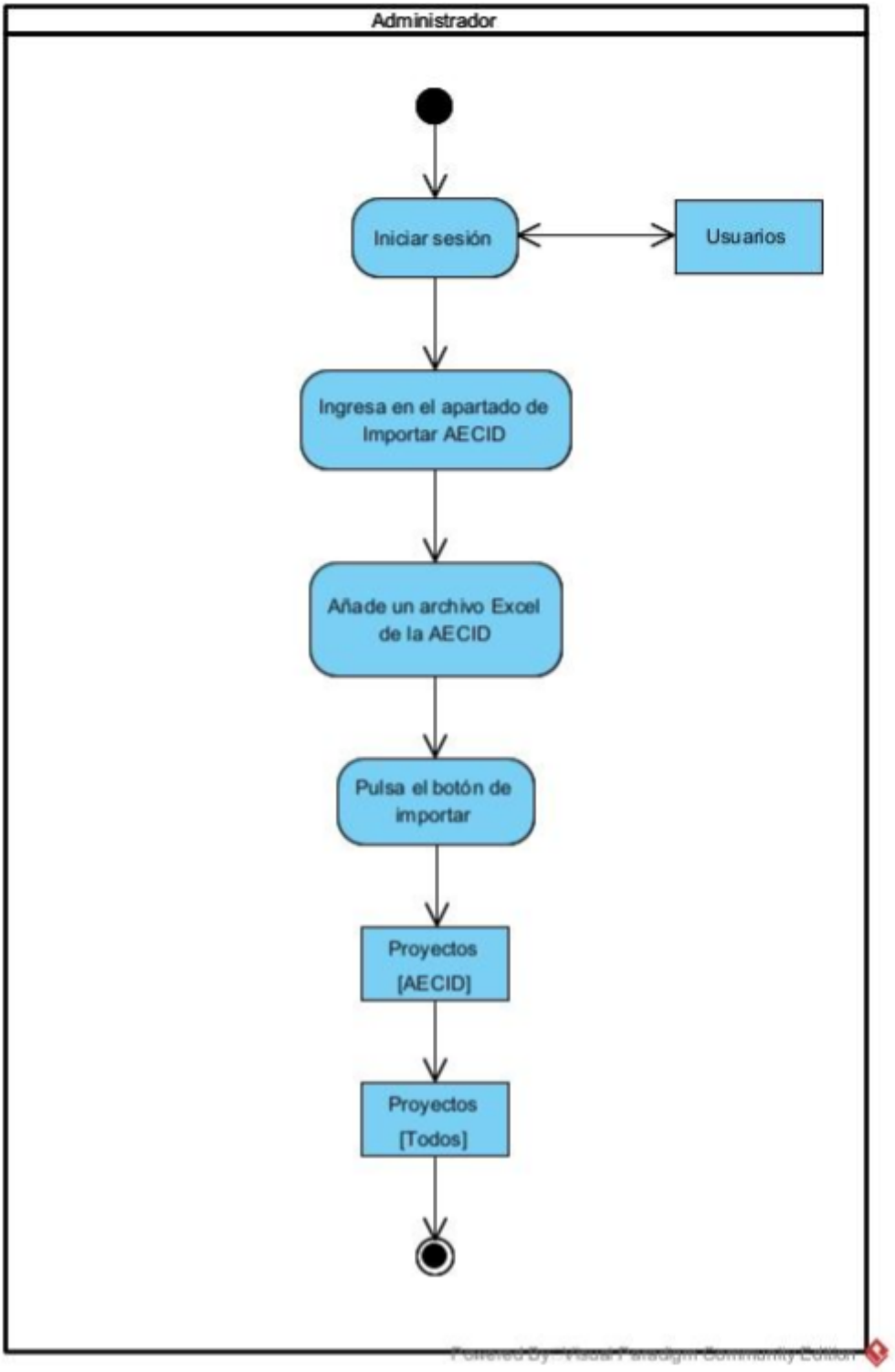

Figura 3.8: Diagrama de actividad del proceso de importación de los proyectos de un archivo Excel de la AECID

3.5 Arquitectura

Respecto a la arquitectura no se ha realizado ninguna modificación, por lo que simplemente comentaré la que utilizó el autor del TFG anterior.

Este decidió, que al tratarse de una renovación, utilizaría una arquitectura dividida principalmente en cuatro capas.

- **Capa de presentación**, referente a la interacción entre el usuario y el sistema, compuesta por la interfaz de usuario y la lógica asociada a la misma. Su principal responsabilidad es mostrar la información al usuario.

- **Capa de negocio**, es la encargada de recibir las peticiones del usuario, procesar la información de entrada y realizar las funciones necesarias para tratar la información.

- **Capa de servicios**, que consiste en la lógica que realiza las funciones principales de la aplicación: procesamiento de datos, implementación de funciones de negocio.

- **Capa de base de datos**, encargada de la persistencia de los datos.

A continuación se muestra el diagrama de despliegue de la solución utilizado por el autor.

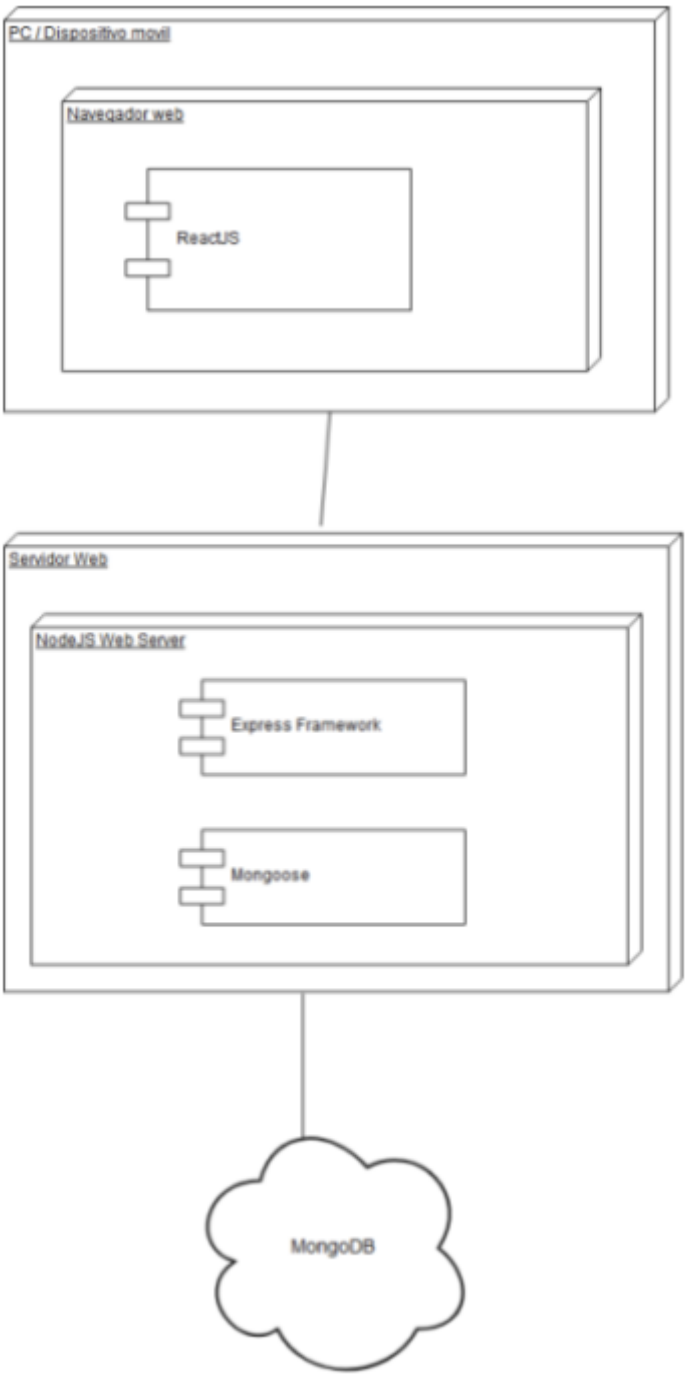

Figura 3.9: Diagrama de despliegue.
Fernandez [2019]

3.6 Diagramas de secuencia

A continuación se muestran dos diagramas de secuencia sobre dos procesos principales de la plataforma: el inicio de sesión y la creación de un nuevo anuncio.

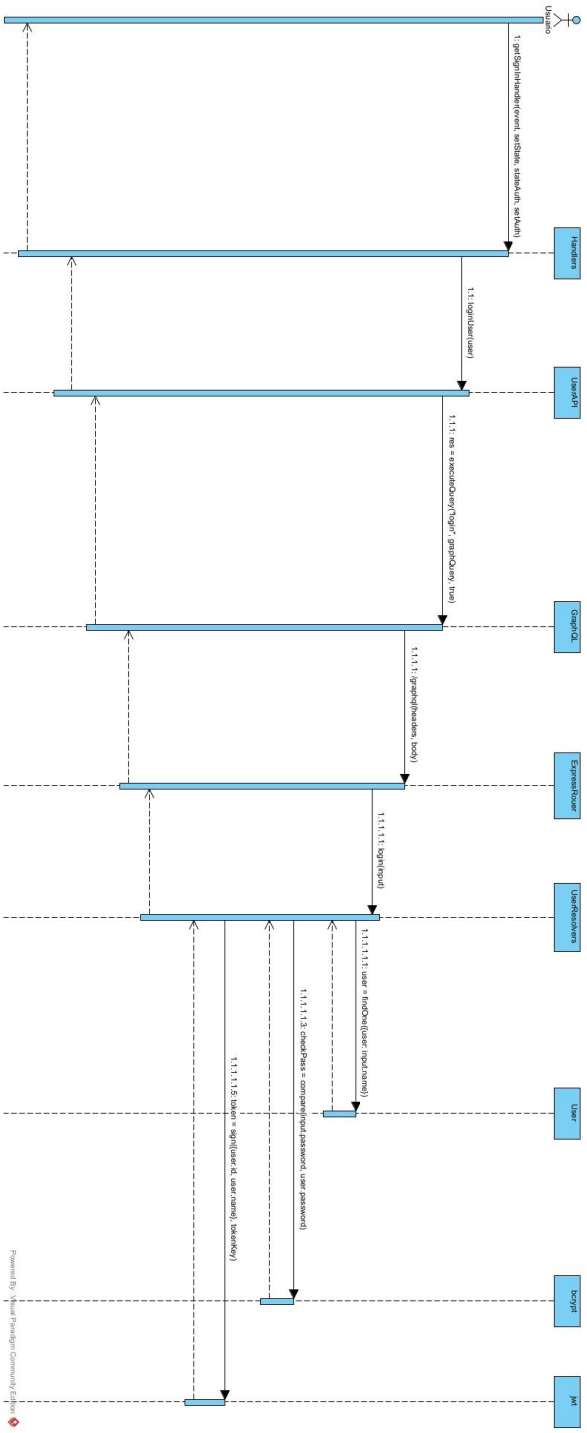

Figura 3.10: Diagrama de secuencia del inicio de sesión

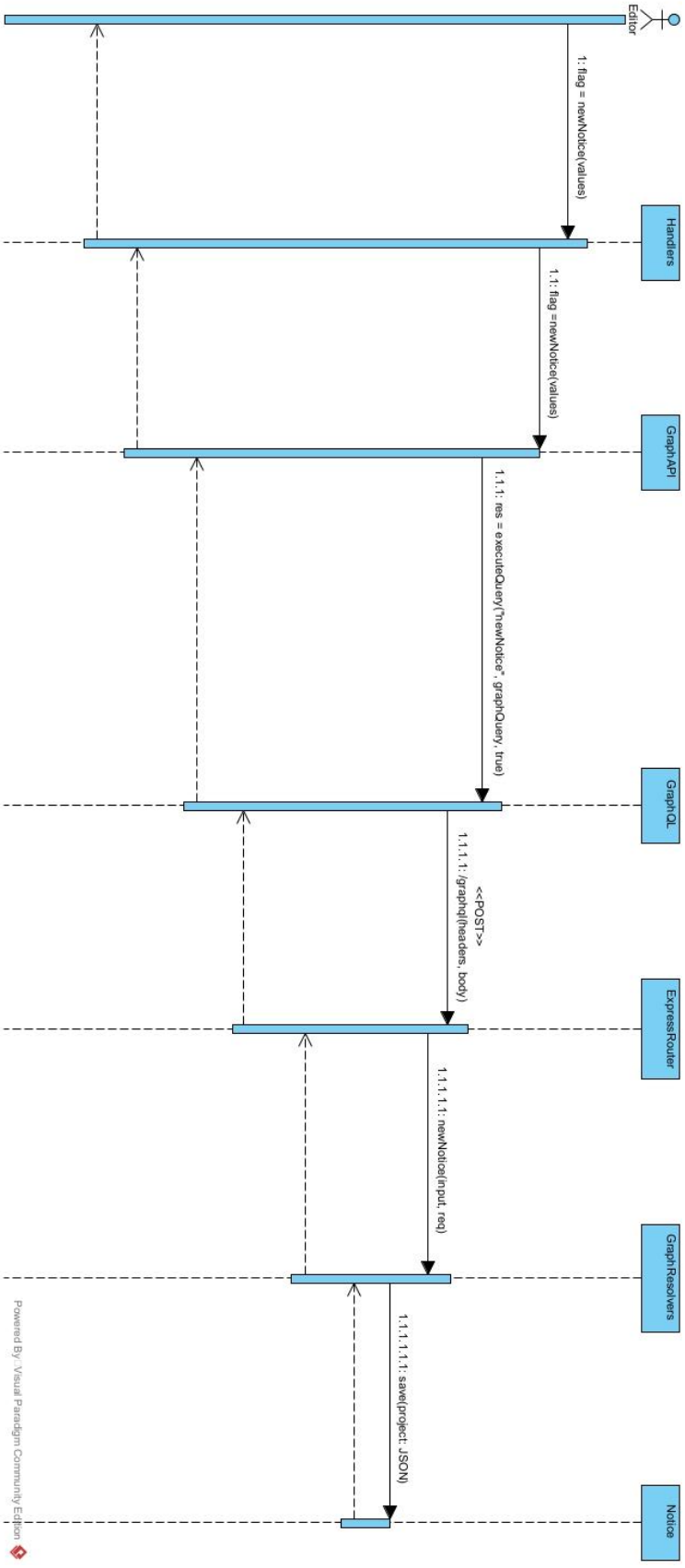

Figura 3.11: Diagrama de secuencia de la creación de un nuevo anuncio

4 Planteamiento del proyecto

4.1 SCRUM

Scrum es un marco de trabajo en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Puede tomarse como conjunto base para definir el proceso de producción que usará un equipo de trabajo o dentro de un proyecto.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

Scrum también se utiliza para resolver situaciones en que no se está entregando al cliente lo que necesita, cuando las entregas se alargan demasiado, los costes se disparan o la calidad no es aceptable, cuando se necesita capacidad de reacción ante la competencia, cuando la moral de los equipos es baja y la rotación alta, cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto.

Los tres roles principales que existen en Scrum son el **Scrum Master,** que es el encargado de aplicar Scrum y gestionar los cambios, el **Product Owner,** el cual debe procurar que el equipo aporte valor al negocio en cuestión y el ***Team*** o **Equipo,** que ejecuta el desarrollo y demás elementos relacionados con él.

El desarrollo se divide en Sprints, periodos comúnmente de entre una y cuatro semanas, esta magnitud es definida en un primer momento por el equipo y debería ser lo más corta posible. Durante cada Sprint, se deberá crear un incremento de software potencialmente entregable, es decir, que pueda ser utilizable o visible para el cliente.

El conjunto de características que forma parte de cada sprint viene del Product Backlog, que es un listado de todas las tareas que se pretenden hacer durante el desarrollo de un proyecto.

Antes de cada Sprint se realizará una reunión llamada *Sprint Planning* en la que el Product Owner identificará cuáles de las tareas del Product Backlog serán las que se realizarán en cada Sprint y se lo dará a conocer al resto del equipo. Entonces, el equipo conversa con el o la Product Owner buscando la claridad y magnitud adecuadas, cumpliendo con los principios INVEST para luego determinar la cantidad de ese trabajo que puede comprometerse a completar durante el siguiente sprint. Además, durante el sprint, nadie puede cambiar el Sprint Backlog, lo que significa que los requisitos están congelados durante el sprint.

4.2 Planificación

Para la realización del proyecto mi decisión ha sido aplicar conceptos de Scrum en el desarrollo del mismo. No era posible aplicar una metodología Scrum completa, puesto que está pensada para realizar un desarrollo en equipo, y en este caso el desarrollo ha sido realizado por una sola persona. Por lo tanto, yo he tenido que desempeñar los roles de Scrum Master, Product Owner y Equipo.

Tampoco se ha respetado completamente la duración de los Sprints, en un principio pensados de dos semanas, debido a que no me ha sido posible dedicar tiempos regulares a la realización del proyecto.

El primer paso fue convertir las demandas exigidas en historias de usuario, las cuales detallaré en el siguiente apartado. Una vez completadas las historias e introducidas en el Product Backlog, se fueron eligiendo cuáles serían realizadas en cada Sprint.

Para la estimación de cada historia la medida utilizada fueron los *Story Points*. Estos son una unidad de estimación que mide la cantidad de esfuerzo que hay que realizar para lograr finalizar una de las tareas. Algunos de los parámetros que se utilizan para realizar las estimaciones son los siguientes:

- Cantidad de trabajo a realizar
- Complejidad de las tareas

- Cualquier riesgo o incertidumbre que pueda comprometer su consecución
- Capacidades técnicas del equipo

De esta forma, se asigna un número determinado de puntos a cada tarea, esto puede variar de un equipo de trabajo a otro, por lo que el primer paso sería decidir qué tipo de tarea sería la que el equipo podría realizar utilizando un *Story Point*, y de esta forma, estimar el esfuerzo necesario para realizar el resto de tareas tomando como referencia el que deberían realizar para finalizar la primera.

4.3 Historias de usuario

Opción de consultar datos de proyectos	
Yo como	Visitante de la web
Quiero	Poder visualizar la información de los proyectos
Para	Consultar los datos relacionados con los proyectos que me interesan
Criterios de aceptación	
Dado	Un proyecto del listado
Cuando	Accedo a él con la opción de ver seleccionada
Entonces	Se mostrarán los datos de este con la edición deshabilitada

Gestión de roles de usuarios	
Yo como	Administrador de la web
Quiero	Poder gestionar los roles de los usuarios
Para	Controlar que pueden hacer y a que tienen acceso estos
Criterios de aceptación	
Dado	Una cuenta de usuario
Cuando	Modifico su rol
Entonces	Las partes a las que puede acceder y las acciones que puede realizar el usuario cambian en función del rol seleccionado

Exportar a Excel datos de un proyecto	
Yo como	Usuario de la web
Quiero	Poder exportar los datos de los proyectos a excel
Para	Almacenar los datos del proyecto de forma externa o trabajar con ellos en otra parte
Criterios de aceptación	
Dado	Un proyecto
Cuando	Pulso el botón de exportar
Entonces	Se generará un archivo excel con los datos del proyecto

Tablón de anuncios	
Yo como	Visitante de la web
Quiero	Poder consultar anuncios sobre la web
Para	Estar informado sobre las últimas novedades de la web
Criterios de aceptación	
Dado	El inicio de la web
Cuando	Pulso el botón de Tablón de anuncios
Entonces	Se mostrarán los anuncios de la web

Gestión de anuncios para el tablón	
Yo como	Editor de la web
Quiero	Poder crear, editar y eliminar anuncios
Para	Informar a los visitantes de la web de las últimas novedades de esta
Criterios de aceptación	
Dado	Un anuncio
Cuando	Pulso el botón de eliminar
Entonces	El anuncio será eliminado
Dado	Un anuncio

Cuando	Pulso el botón de editar
Entonces	Se podrán editar los campos del anuncio
Dado	El tablón de anuncios
Cuando	Pulso el botón de nuevo anuncio
Entonces	Aparecerá el formulario de creación de nuevo anuncio

Gestión del slide de imágenes en el inicio	
Yo como	Editor de la web
Quiero	Poder añadir imágenes para que se muestren en un carrusel en la página de inicio
Para	Que los visitantes de la web puedan verlas nada más entren a la web
Criterios de aceptación	
Dado	El gestor de imágenes del carrusel
Cuando	Pulso el botón de eliminar
Entonces	La imagen se eliminará del carrusel del inico
Dado	El gestor de imágenes del carrusel
Cuando	Añado una imagen
Entonces	Esta se mostrará en el carrusel del inicio

Añadir más campos seleccionables a la creación de gráficas	
Yo como	Visitante de la web
Quiero	Poder seleccionar varios campos en el apartado de gráficas
Para	Poder ver las gráficas que se generan con los datos de cada uno de los campos elegidos
Criterios de aceptación	
Dado	El apartado de gráficas
Cuando	Selecciono dos campos
Entonces	Se generará una gráfica en base a los dos campos seleccionados

Importar datos de la AECID	
Yo como	Usuario editor de la web
Quiero	Poder importar los proyectos que la AECID permite descargar en formato Excel desde su web
Para	Poder acceder y trabajar con estos datos más cómodamente desde la web, sin la necesidad de tener que añadirlos manualmente de uno en uno
Criterios de aceptación	
Dado	Un archivo excel con el formato de la AECID
Cuando	Se pulsa el botón de importar
Entonces	Se añadirán todos los proyectos y datos compatibles del excel a la base de datos de la web

Añadir diferentes tipos de gráficas	
Yo como	Usuario de la web
Quiero	Poder ver diferentes tipos de gráficas
Para	Poder ver los datos en diferentes formatos
Criterios de aceptación	
Dado	El selector de tipo de gráfica
Cuando	Elijo uno de los tipos
Entonces	Se generará una gráfica del tipo seleccionado

Creación y gestión de anuncios internos	
Yo como	Administrador de la web
Quiero	Poder crear y modificar anuncios internos en la web
Para	Poder informar de las novedades a los usuarios registrados de la web
Criterios de aceptación	
Dado	Un anuncio interno
Cuando	Pulso el botón de eliminar
Entonces	El anuncio será eliminado
Dado	Un anuncio interno

Cuando	Pulso el botón de editar
Entonces	Se podrán editar los campos del anuncio
Dado	El apartado dirección de proyectos
Cuando	Pulso el botón de nuevo anuncio
Entonces	Se podrá crear un nuevo anuncio interno

Crear apartado "dirección de proyectos"	
Yo como	Usuario registrado de la web
Quiero	Poder ver los anuncios internos que cree el administrador
Para	Estar informado de las últimas novedades a nivel interno de la web
Criterios de aceptación	
Dado	El botón del apartado dirección de proyectos
Cuando	Lo pulso habiendo iniciado sesión en la web
Entonces	Aparecerán todos los anuncios internos de la web

Añadir nuevas formas de buscar proyectos	
Yo como	Usuario de la web
Quiero	Poder buscar proyectos a partir de diferentes campos
Para	Que me sea más sencillo encontrar el proyecto que busco
Criterios de aceptación	
Dado	El selector de tipo de búsqueda
Cuando	Elijo un valor de la lista
Entonces	Se mostrará el listado de proyectos ordenados por el campo elegido

4.4 Otras tareas a realizar

Además de las historias de usuario arriba mencionadas, ha sido necesario realizar otros cambios como solución de errores o correcciones menores, que ya existían en la web o han ido surgiendo mientras se revisaban las nuevas funcionalidades implementados. A continuación listaré estos cambios.

- Algunos campos de los proyectos aparecen sin nombre.
- Mover el cuadro de inicio de sesión a una ventana flotante a la que se pueda acceder desde cualquier parte de la web, y no solo desde el inicio como hasta ahora
- Revisar el funcionamiento de las gráficas. Algunas aparecen con errores de escala.
- Añadir “rueda de carga” mientras se realizan peticiones que tardan algo de tiempo en realizarse como la generación de gráficas por ejemplo.
- Corrección de errores en la creación de anuncios.

5 Ejecución del proyecto

5.1 Estructura del proyecto

La estructura de archivos del proyecto está dividida en en dos carpetas principales y varios archivos que detallaré a continuación:

- La carpeta **public**, que contiene los archivos e imágenes de la web, además del archivo compilado bundle.js que genera webpack y el archivo index.html, el cual es el punto de inicio de la aplicación.
- La carpeta **src,** en la que se encuentran a su vez las carpetas **app**, que contiene todos los componentes y archivos necesarios de la aplicación React, la carpeta **models**, que define los modelos de objetos que se utilizarán como plantilla de los datos que se almacenarán en la base de datos, y por último la carpeta **server**, que contiene los archivos de NodeJS, ExpressJs, Mongoose y GraphQL, es decir, los archivos que se ejecutarán en el servidor.
- Varios archivos de configuración, como **package.json**, que contiene las dependencias de npm o **webpack.config.js**, que contiene los parámetros de configuración de Webpack.

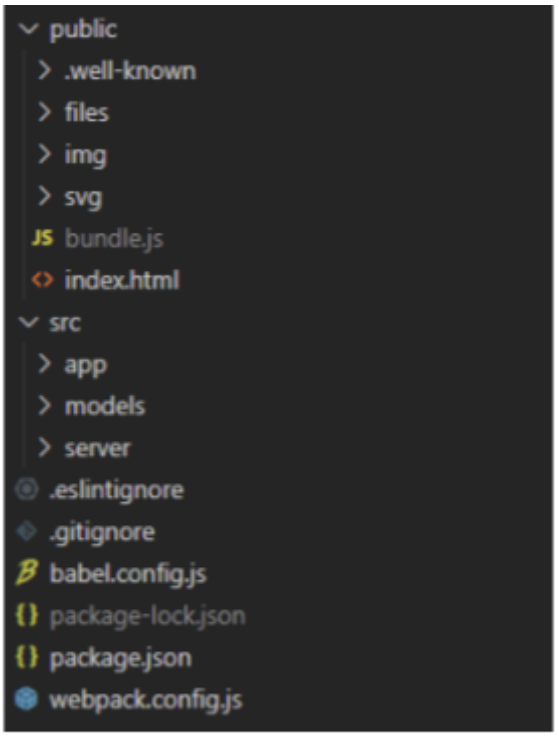

Figura 5.1: Estructura de archivos

5.2 Desarrollo del proyecto

Como expliqué anteriormente, en la planificación, el proyecto ha sido realizado en Sprints, pensados para ser realizados en dos semanas cada uno, pero debido a la imposibilidad de realizarlos en un horario regular cada día no se han respetado completamente estos tiempos. También, dependiendo del tiempo disponible para la realización del proyecto, en alguno de los Sprints se pudo asignar más o menos carga de trabajo que en el resto, por lo que no existiría un equilibrio de carga de trabajo entre los diferentes Sprints.

Finalmente el desarrollo del proyecto se ha realizado en siete Sprints, los cuáles detallaré a continuación.

5.2.1 Primer Sprint

El primer paso fue seleccionar qué historias de usuario y tareas se iban a realizar en el próximo Sprint para ajustarse al plazo de dos semanas. En este caso se seleccionaron primeramente tres historias de usuario

El siguiente paso fue asignar los *story points* estimados a cada tarea, en función de la complejidad y extensión de cada una de las historias, quedando de la siguiente forma:

- *Opción de consultar datos de proyectos*: 3 Story points
- *Gestión de roles de usuarios*: 5 Story points
- *Exportar a Excel datos de un proyecto*: 8 Story points.

Como se puede observar, la historia que más complejidad tenía es la de *Exportar a Excel datos de un proyecto*, puesto que era necesaria una librería de React que permitiera convertir los datos del proyecto en formato json a un archivo excel. Las otras dos historias eran más sencillas de realizar, por lo que le fueron asignados menos Story points.

Luego se estimó el tiempo que llevaría realizar cada una de las tareas, el cual se muestra en la gráfica de trabajo que aparece más adelante.

Después de iniciado el Sprint se decidió añadir una tarea más a este, puesto que sobraba tiempo del Sprint habiendo finalizado ya las tareas asignadas a este. La tarea añadida al Sprint fue:

- *Corregir campos de los proyectos*

En la siguiente figura se muestra el progreso de trabajo a lo largo del sprint.

Nombre de la tarea	Horas estimadas	Horas reales	Diferencia de horas	Story points	Día de inicio	Día de finalización	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Opción de consultar datos de proyectos	10	10	0	5	1	4	4		3	3										
Gestión de roles de usuarios	6	5	1	3	6	7						2	3							
Exportar a excel datos de un proyecto	16	16	0	8	8	11								4	4	4	4			
Corregir campos de los proyectos	3	3	0		12	12												3		

Figura 5.2: Gráfica de trabajo del primer Sprint

Como se observa en la gráfica, la estimación quedó por debajo del plazo de dos semanas aun habiendo añadido una tarea más, esto fue debido a que no sabía si me sería posible trabajar en el proyecto durante todos los días restantes del Sprint, así que mi decisión fue no añadir más trabajo a este.

5.2.2 Segundo Sprint

Para este segundo Sprint se decidieron añadir cuatro tareas, las cuales serían las siguientes:

- *Tablón de anuncios*
- *Gestión de anuncios para el tablón*
- *Gestión del slide de imágenes en el inicio*
- *Mover el cuadro de inicio de sesión a una ventana flotante*

Seguidamente se asignaron *Story points* a cada una de las historias de usuario, además del tiempo estimado que llevaría realizarlas. Todos estos datos aparecen en la gráfica de trabajo del Sprint.

Nombre de la tarea	Horas estimadas	Horas reales	Diferencia de horas	Story points	Día de inicio	Día de finalización	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Tablón de anuncios	24	24	0	12	1	6	3		6	7	4	4								
Gestión de anuncios para el tablón	16	17	-1	8	7	10							2	5	6	4				
Gestión del slide de imágenes en el inicio	12	12	0	6	11	14											4	3	3	2
Mover el cuadro de inicio de sesión a una ventana flotante	4																			

Figura 5.3: Gráfica de trabajo del segundo Sprint

En la gráfica se puede observar que a parte de haber subestimado las horas de la tarea *Gestión de anuncios para el tablón* la última tarea del Sprint no pudo llegar a realizarse finalmente dentro del plazo de dos semanas. Por lo que esta tarea se movió al siguiente Sprint.

5.2.3 Tercer Sprint

En el tercer Sprint se añadieron dos tareas a realizar además de la que había quedado pendiente del anterior Sprint, quedando finalmente las siguientes tareas a realizar en las siguientes dos semanas:

- *Mover el cuadro de inicio de sesión a una ventana flotante*
- *Añadir más campos seleccionables a la creación de gráficas*
- *Revisar funcionamiento de las gráficas*

En este caso el tiempo disponible fue más bajo de lo habitual, además de que viendo que en el Sprint anterior se añadió más carga de trabajo de la que fue posible realizarse al final, mi decisión fue asignar pocas tareas, que pudieran estimarse con pocas horas para este Sprint.

Por lo tanto después de asignar las diferentes estimaciones a cada una de las tareas y pasadas las dos semanas del Sprint la gráfica de trabajo quedaría de la siguiente forma:

Nombre de la tarea	Horas estimadas	Horas reales	Diferencia de horas	Story points	Día de inicio	Día de finalización	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Mover el cuadro de inicio de sesión a una ventana flotante	4	4	0		1	3	2		2											
Añadir más campos seleccionables a la creación de gráficas	12	13	-1	6	6	10						3	2		4	4				
Revisar funcionamiento de las gráficas	8	6	2		11	14											2		2	2

Figura 5.4: Gráfica de trabajo del tercer Sprint

En la gráfica se observa que una de las tareas se subestimó una hora por debajo del tiempo que finalmente llevó, mientras que otra de las tareas se sobreestimó, finalizando esta dos horas antes de lo estimado.

5.2.4 Cuarto Sprint

En este Sprint solamente se añadió una única tarea a realizar debido a su complejidad, la cual es la siguiente:

- *Importar datos de la* AECID

Esta tarea consiste en leer los datos de un fichero Excel con el formato específico que la AECID permite descargarlo desde su web. Requiriendo entonces localizar cuáles de los campos del archivo se almacenan en la base de datos de la web y con cuál de ellos se corresponden.

En este Sprint, la gráfica de trabajo quedaría de esta forma:

Nombre de la tarea	Horas estimadas	Horas reales	Diferencia de horas	Story points	Día de inicio	Día de finalización	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Importar datos de la AECID	40	40	0	20	1	13	4		3	4	5	4		2	3	4	2	4	5	

Figura 5.5: Gráfica de trabajo del cuarto Sprint

Finalmente se realizó la tarea un día antes de la terminación del plazo del Sprint, pero debido al poco tiempo sobrante no fue añadida ninguna tarea más a realizar dentro de este Sprint, dándolo entonces por finalizado.

5.2.5 Quinto Sprint

Al quinto Sprint le fueron asignadas tres tareas, siendo entonces la lista de tareas a realizar en el Sprint la siguiente:

- *Añadir diferentes tipos de gráficas*
- *Mejorar la búsqueda y selección de proyectos a la hora de ver o editar*
- *Añadir "rueda" de carga a varios procesos*

Para la segunda de las tareas se decidió sustituir el selector de proyecto, que cargaba un listado con los identificadores de todos los proyectos presentes en la base de datos por un cuadro de texto del que se desplegará una lista que irá filtrando los proyectos según lo que se vaya escribiendo.

La gráfica de trabajo de este Sprint quedaría de la siguiente forma:

Nombre de la tarea	Horas estimadas	Horas reales	Diferencia de horas	Story points	Día de inicio	Día de finalización	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Añadir diferentes tipos de gráficas	20	17	3	10	1	6	3	4		2	5	3								
Mejorar la búsqueda y selección de proyectos a la hora de ver o editar	12	12	0		7	11							2	4	4		2			
Añadir "rueda" de carga a varios procesos	6	4	2		13	14													3	1

Figura.5.6: Gráfica de trabajo del quinto Sprint

En este Sprint se acabaron las tareas el último día, pero dos de ellas se terminaron antes de lo previsto, haciéndolo en total cinco horas antes de lo estimado.

5.2.6 Sexto Sprint

Para el sexto Sprint se escogieron dos de las tareas restantes del Product Backlog, las cuales son las siguientes:

- *Crear apartado "dirección de proyectos"*
- *Creación y gestión de anuncios internos*

Este apartado es muy similar al ya creado anteriormente "*Tablón de anuncios*", con la diferencia de que en este caso, los anuncios solo serán visibles para los usuarios que accedan con cuenta de usuario a la web. Por lo que el desarrollo de esta parte se basó en añadir un campo público a los anuncios, además de la posibilidad de adjuntar un archivo al anuncio.

Nombre de la tarea	Horas estimadas	Horas reales	Diferencia de horas	Story points	Día de inicio	Día de finalización	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Crear apartado "dirección de proyectos"	16	16	0	8	1	7	2	2	2	4	4		2							
Creación y gestión de anuncios internos	24	22	2	12	8	14								4	4		6	3	2	3

Figura 5.7: Gráfica de trabajo del sexto Sprint

El Sprint finalizó con la primera tarea estimada correctamente mientras que en la segunda sobraron 2 horas respecto a lo que se había estimado.

5.2.7 Séptimo Sprint

En este último Sprint se cogieron las tareas restantes que quedaban en el *Product Backlog*, las cuales eran:

- *Añadir nuevas formas de buscar proyectos*
- *Corrección de errores en la creación de anuncios*

Para la primera tarea se decidieron añadir a la selección de filtrado de proyectos las opciones de País, Año y Título, facilitando de este modo la búsqueda del proyecto que se desea consultar o editar. La segunda de estas tareas se añadió al final, debido a que se encontraron fallos en la creación de anuncios.

Nombre de la tarea	Horas estimadas	Horas reales	Diferencia de horas	Story points	Día de inicio	Día de finalización	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Añadir nuevas formas de buscar proyectos	8	8	0	4	1	4	3		3	2										
Corrección de errores en la creación de anuncios	4	4	0	12	6	7						2	2							

Figura 5.8: Gráfica de trabajo del séptimo Sprint

Como se puede observar en la gráfica sobró la mitad de tiempo del Sprint, esto se debió a que las dos únicas tareas que restaban por finalizar eran bastante

cortas, por lo que no había trabajo suficiente para rellenar las dos semanas de este último Sprint.

5.3 Cambios en la interfaz de la aplicación

En este apartado se mostrarán algunos de los cambios que ha habido en la interfaz de la aplicación debido a las funcionalidades añadidas y cambios realizados respecto a la versión anterior de esta.

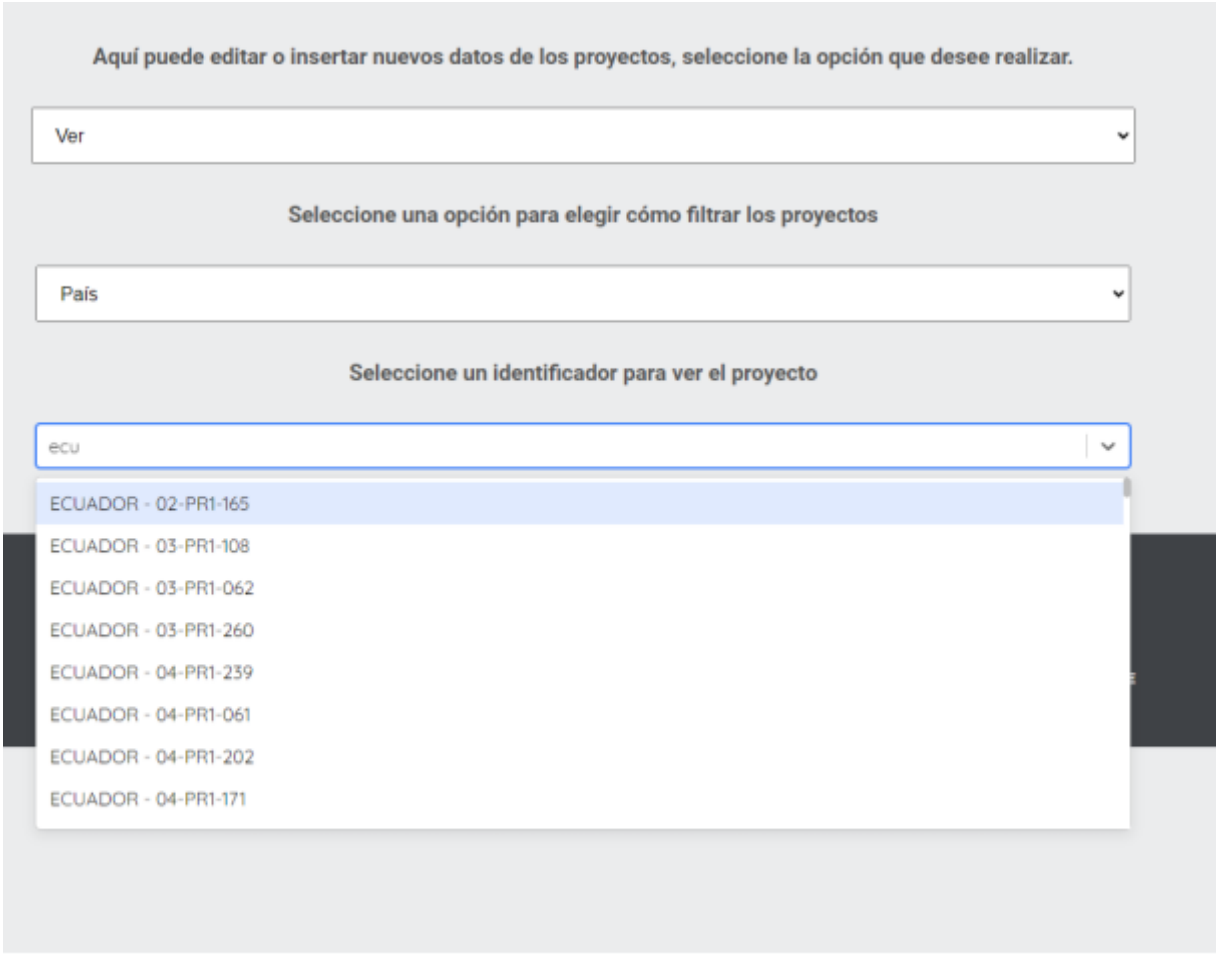

Figura 5.9: Nuevo selector de proyectos

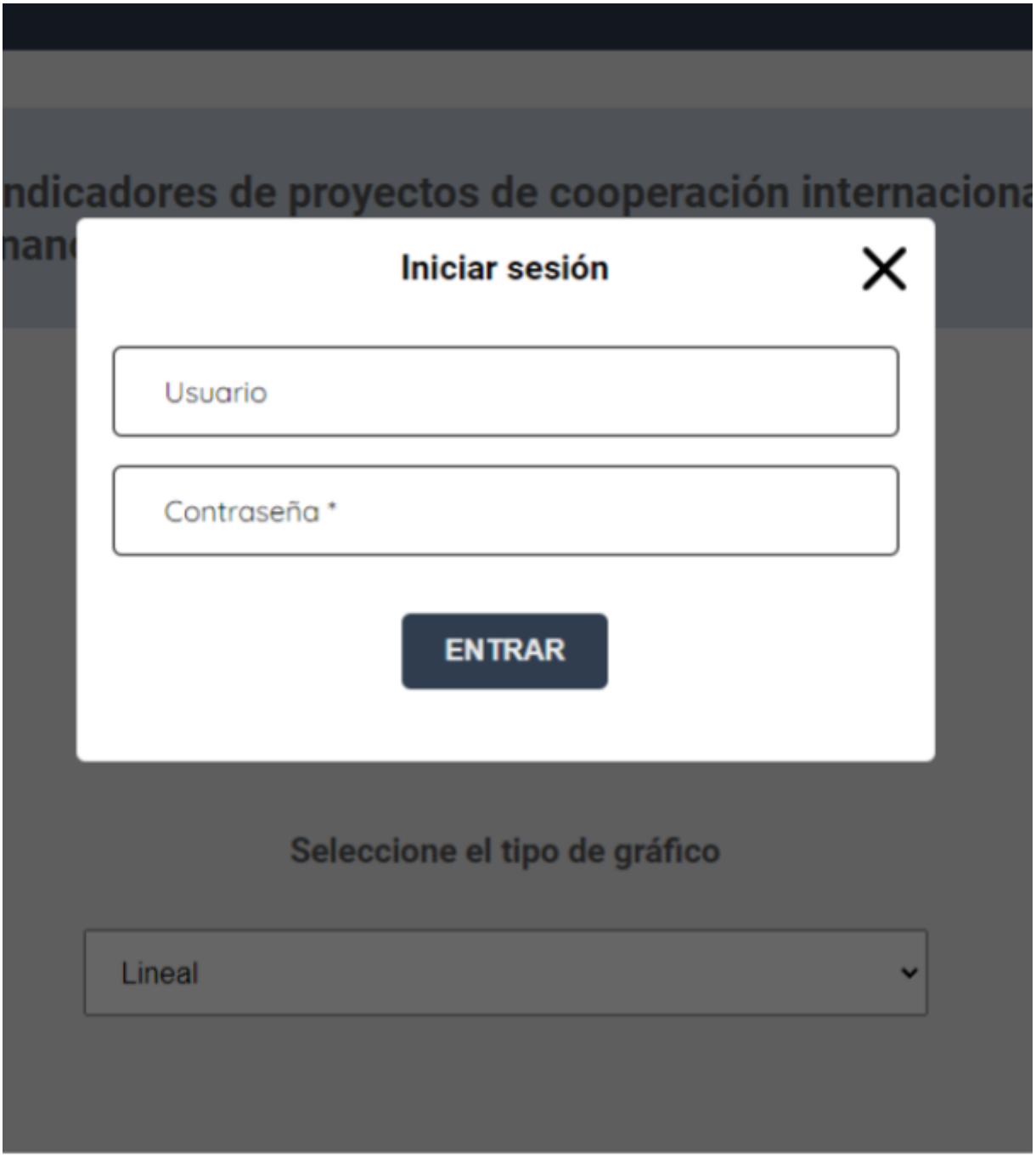

Figura 5.10: Cuadro de inicio de sesión

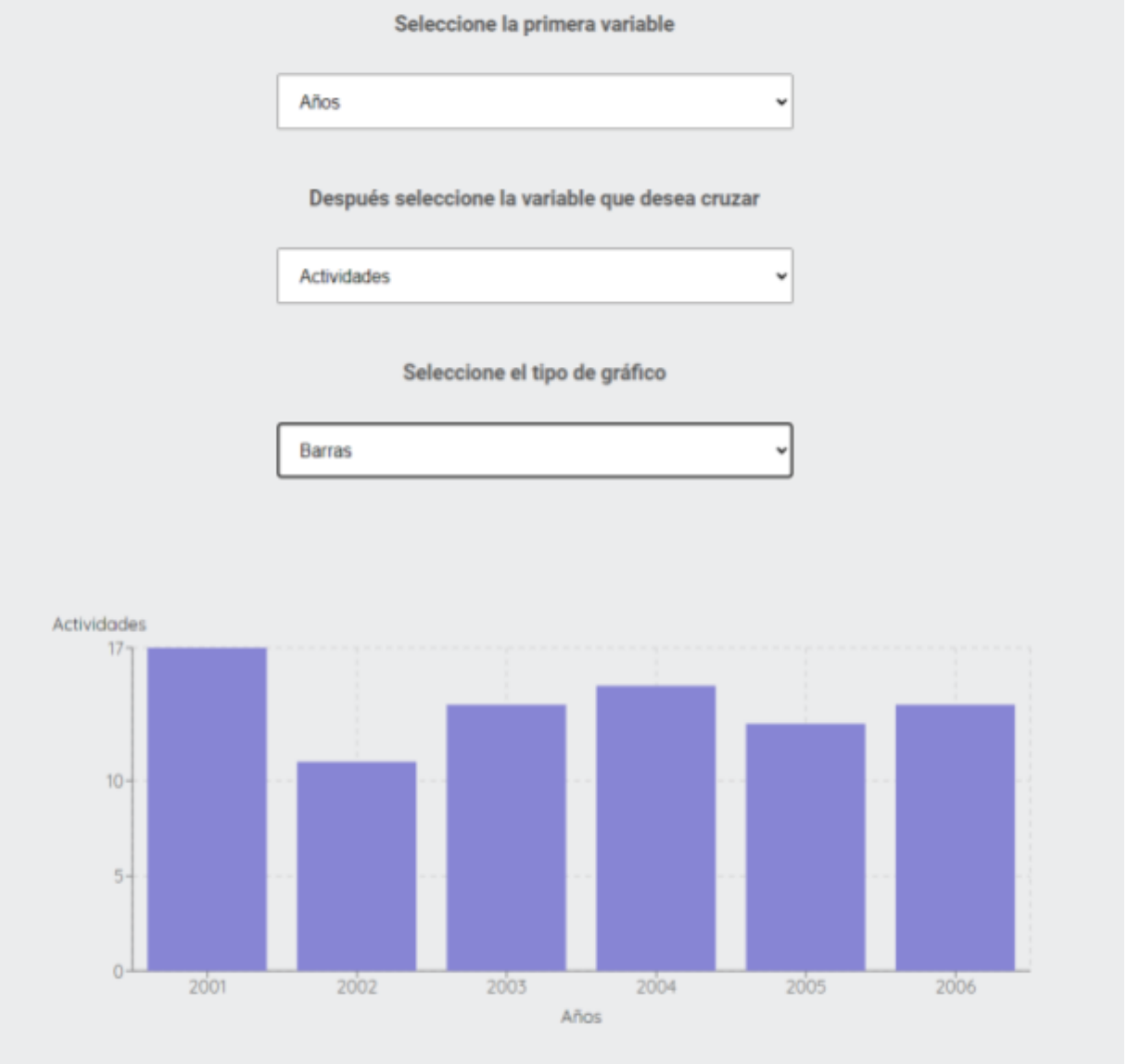

Figura 5.11: Nuevos tipos de gráficas

Figura 5.12: Tablón de anuncios

Figura 5.13: Ventana de inserción de un anuncio

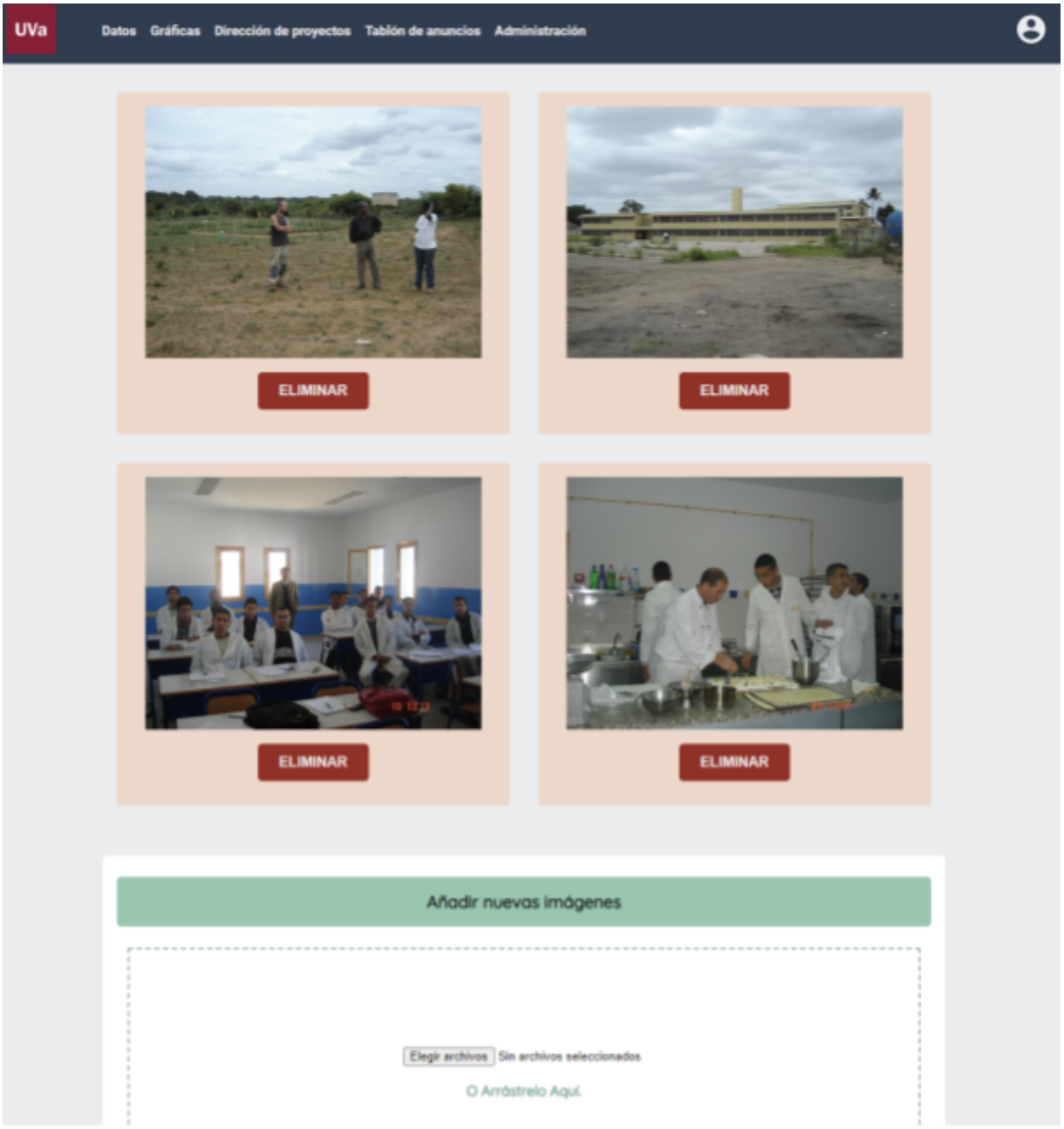

Figura 5.14: Ventana de gestión del carrusel

6 Conclusiones

En este último apartado detallaré las conclusiones a las que he llegado a lo largo del desarrollo del proyecto.

El objetivo del Trabajo Fin de Grado era añadir características y corregir errores de una web desarrollada en otro trabajo de fin de grado. Como el trabajo consistió en modificar una web ya existente, las tecnologías que utilizaba ya estaban fijadas. Debido a esto, el primer paso fue aprender a utilizar estas tecnologías, puesto que no había tenido la oportunidad de trabajar antes con ninguna de ellas.

Una vez comprendido el funcionamiento, me han parecido un conjunto de tecnologías muy cómodo de utilizar, y no muy complicadas, a excepción de NodeJs, que podría ser la más compleja de todas.

MongoDB ha sido la primera base de datos NoSQL que he utilizado, y veo bastante cómoda la forma que tiene de almacenar los datos en un formato estilo JSON, permitiendo añadir nuevos campos sin necesidad de modificar la estructura de la "tabla", como si que habría que hacer en una base de datos SQL.

A día de hoy el stack MERN se encuentra entre los más utilizados junto al stack MEAN, en el que el único cambio sería el de React por Angular, por lo que aprender a desarrollar con cualquiera de los dos stacks hace que ya tengas aprendidas las tecnologías de NodeJS, ExpressJS y MongoDB, entonces, en el caso de cambiar entre un stack y otro solo sería necesario aprender la tecnología de la parte de Frontend.

Al ser tecnologías modernas bastante utilizadas a día de hoy, tener conocimientos sobre ellas será algo bastante útil en el mercado laboral, puesto que según la tendencia actual apunta hacia que se seguirán utilizando durante bastantes años, haciendo que cada vez más empresas las utilicen para el desarrollo de sus proyectos.

Otra de las ventajas de estos stacks es que solo sería necesario conocer JavaScript para desarrollar tanto la parte de Frontend como la de Backend, siendo bastante cómoda la posibilidad de poder escribir todo el código de la aplicación en el mismo lenguaje.

Bibliografía

José Miguel Fernández. Sáenz de Navarrete *Diseño e implantación de un entorno web con NodeJS*. Universidad de Valladolid, 2019

Larman, C. *UML y Patrones. Introducción al Análisis y Diseño Orientado a Objetos y al Proceso Unificado*. Prentice Hall, 2002.

Booch, G., Jacobson, I., Rumbaugh, J. *El Lenguaje Unificado de Modelado. Guía del usuario*. Addison-Wesley/Diaz de Santos, 2º edición, 2005.

George Ornbo. (25 de marzo de 2014). *Writing cross-platform Node.js*. https://shapeshed.com/writing-cross-platform-node/

Krissanawat Kaewsanmuang. (14 de enero de 2019). *React file upload: proper and easy way, with NodeJS!*. https://programmingwithmosh.com/javascript/react-file-upload-proper-server-side-nodejs-easy/

Asher Nguyen. *Documentación de react-id-swiper*. https://react-id-swiper.ashernguyen.site/doc/get-started

Recharts Group. *Documentación de Recharts*. https://recharts.org/en-US/api

Jed Watson. *Documentación de react-select*. https://react-select.com/home

Proyectos Ágiles. *¿Qué es SCRUM?* https://proyectosagiles.org/que-es-scrum/

Javier Martínez Solera. (4 de julio de 2020). *¿Qué es MERN Stack?* https://jmsolera.com/que-es-mern/

Figura arquitectura MERN Stack. https://www.mongodb.com/mern-stack

React. *Documentación de React*. https://es.reactjs.org/docs/getting-started.html

Node.js. *Documentación de Node.js*. https://nodejs.org/es/docs/

Victor Valencia Rico. (21 de mayo de 2020). *¿Qué es Webpack?* https://medium.com/@victor.valencia.rico/qu%C3%A9-es-webpack-75cb56559759

Marías Hernandez (9 de marzo de 2021). *¿Qué es Babel?* https://www.freecodecamp.org/espanol/news/que-es-babel/

John Serrano (15 de junio de 2020). *Aprende a crear rutas con React Router.* https://johnserrano.co/blog/aprende-a-crear-rutas-con-react-router

Jhuleidy Acosta (29 de junio de 2021). *GraphQL: Qué es y qué ventajas ofrece* https://openwebinars.net/blog/graphql-que-es-y-que-ventajas-ofrece/

Jesús Lucas (9 de septiembre de 2019). *Qué es NodeJS y para qué sirve.* https://openwebinars.net/blog/que-es-nodejs/

MDN Contributors. (8 de diciembre de 2020). *Introducción a Express/Node.* https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction

MongoDB. *¿Qué es MongoDB?* https://www.mongodb.com/es/what-is-mongodb

Chiyana Simões. (20 de julio de 2021). *JavaScript: ¿Qué es npm?* https://www.itdo.com/blog/javascript-que-es-npm/