UNIVERSITAT DE BARCELONA

Treball de Fi de Grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

IMPLEMENTACIÓN DE LA APLICACIÓN WEB NEWS OF THE WORLD

Víctor Herrera Malato

Director: Eloy García Marcos
Realitzat a: Departament de Matemàtiques i Informàtica

Barcelona, 13 de juny de 2022

Resumen (Castellano)

Como todos sabemos, estamos en la era de la información, donde con sólo un clic podemos encontrar miles de noticias algunas verídicas y otras no, las bien conocidas fake news. Muchas veces, las personas necesitamos una página de confianza donde leer información que, a nuestro juicio, es verídica y de rigor.

Por eso, he creído conveniente implementar la aplicación News of the World. Esta aplicación se trata de una página web, donde podemos encontrar artículos categorizados en seis grandes blogs de distinta índole, implementada para ser accedida a través de cualquier navegador web de cualquier ordenador. Para adentrarnos en este proyecto, hemos llevado a cabo una primera búsqueda de los antecedentes sobre las noticias y he realizado un estudio para saber cómo debería ser la estructura más idónea para resolver esta necesidad. Además, se trata de una aplicación libre, donde cualquier persona del mundo puede consultar el código, descargarlo, modificarlo, realizar propuestas de mejora y ejecutarlo sólo accediendo al repositorio de Github donde se encuentra: https://github.com/realBikto/thenews.

En esta memoria mostraremos la metodología, diseño e implementación de una primera versión de la aplicación que cumpla con las principales características descritas, así como la manera de ejecutar la aplicación.

Resum (Català)

Com bé sabem tots, som a l'era de la informació, on amb només un clic podem trobar milers de notícies algunes de verídiques i d'altres no, les ben conegudes *fake news*. Moltes vegades, les persones necessitem una pàgina de confiança on llegir informació que, des del nostre parer, és verídica i de rigor.

Per això, he cregut convenient implementar l'aplicació "News of the World". Aquesta aplicació es tracta d'una pàgina web, on podem trobar articles categoritzats en sis grans blocs de diferent caràcter, implementada per a ser accedida a través de qualsevol navegador web de qualsevol ordinador. Per poder endinsar-nos en aquest projecte, hem dut a terme una primera recerca dels antecedents envers les notícies i he realitzat un estudi per saber com hauria de ser l'estructura més idònia per resoldre aquesta necessitat. A més, es tracta d'una aplicació lliure, on qualsevol persona del món pot consultar el codi, descarregar-lo, modificar-lo, fer propostes de millora i executar-lo només accedint al repositori de Github on es troba: https://github.com/realBikto/thenews.

En aquesta memòria, doncs, mostrarem la metodologia, el disseny, la implementació i la manera d'executar una primera versió de l'aplicació que compleixi amb les principals característiques descrites.

Abstract (English)

As we all know, we are in the information era, where with just one click we can find thousands of news, some truthful and others not, the well-known fake news. Often, people need a trusted page where they can read information that, in their opinion, is truthful and accurate.

For this reason, I have decided to implement the News of the World application. This application is a web page, where we can find articles categorized in six major blogs of different natures, implemented to be accessed through any web browser on any computer. To get into this project, we have performed a first search of the background on the news and I have made a study to know how it should be the most suitable structure to solve this need. In addition, it is an open source application, where anyone in the world can consult the code, download it, modify it, make suggestions for improvement and run it just by accessing the Github repository where it is located: https://github.com/realBikto/thenews.

In this project we will show the methodology, design and implementation of a first version of the application that meets the main features described, as well as how to run the application.

INDEX

1. Introducción

El derecho de acceso a la información resulta esencial para el correcto funcionamiento de las sociedades y para el bienestar personal. La libertad de información, o el derecho a la misma, es parte integrante del derecho fundamental de libertad de expresión, expuesto en la Declaración Universal de Derechos Humanos de 1948. Este derecho nos permite no sólo estar informados de la actualidad sino también ser capaces de aprender, desarrollar opiniones, tomar decisiones y ser críticos cuando sea necesario.

Los desarrollos tecnológicos nos han permitido, a largo de los años, estar informados mediante prensa escrita, en un principio, informativos radiofónicos o televisivos, más tarde, y, últimamente, es más frecuente el acceso a noticieros digitales. Vivimos unos tiempos en los que cualquier persona de los países tecnológicamente avanzados tiene acceso a un teléfono móvil y la gran mayoría a un ordenador, con conexión a internet. Estas tecnologías son, a día de hoy, un acceso permanente a la información, ya sea a través de las noticias que podemos consultar en los diferentes navegadores o aplicaciones, en las notificaciones configurables que nos permiten mantenernos al día sobre cualquier novedad de algún tema de interés, o incluso el contacto permanente con otras personas a través de aplicaciones de mensajería instantánea o en las redes sociales.

Sin embargo, en los últimos años, tal y como ciertas asociaciones internacionales de periodistas han denunciado, existe cierto declive en la ética periodística de determinados grupos o consorcios informativos, en países de todo el mundo, predominando los intereses corporativos o políticos a la hora de informar. La accesibilidad de las nuevas tecnologías provoca que aquellas organizaciones que tienen más poder se adueñen de los medios de comunicación, como los diarios, para hacer llegar al ciudadano noticias sesgadas o claramente politizadas, lo que puede ser más beneficioso para esa organización o sus afines.

El propósito de este trabajo es el desarrollo de un aplicación web que permita compartir y verificar noticias de diversos ámbitos, libre de presiones de organizaciones y con total libertad de expresión.

1.1. Motivación

Hay varios factores que me han motivado a la realización de un diario electrónico. El primero es el propio derecho de acceso a una información veraz y rigurosa. A título personal, debido a que leo muchos diarios, he podido comprobar personalmente que a veces es preciso leer la misma noticia en dos diarios diferentes, puesto que según el signo político con el cual se sienta más identificado el noticiero, una misma noticia puede cambiar su punto de vista y, del mismo modo, las conclusiones que puedes extraer de ella. Cansado de esta situación, que atenta contra el derecho fundamental del acceso a la información y contra la ética del rigor periodístico, pretendo crear un noticiario en el que no haya ninguna influencia de los agentes políticos y sociales más comunes, y que los artículos sean información veraz, de modo que las noticias representen los hechos tal cual han acontecido, sin retoques ni juicios de valor.

Otro de los factores que me ha impulsado a desarrollar este trabajo es que, desde el punto de vista académico que nos ocupa y con vistas a un desarrollo profesional en un ámbito privado, la creación de este diario electrónico, me permite aprender tecnologías que no han sido aplicadas durante el grado pero que son muy utilizadas y conocidas en el mundo laboral. Para hacer el proyecto, he decidido investigar y utilizar tecnologías que no he visto durante mis años en la Universidad pero que sí he visto que son utilizadas en el ámbito laboral o empresarial, aprovechando, de este modo, para estar más preparado de cara a un futuro profesional exitoso.

1.2. Objetivos

El principal objetivo del proyecto es desarrollar una página web que sea accesible desde cualquier ordenador y desde cualquier navegador siguiendo los principios de una aplicación usable, en la que la experiencia del usuario sea la más óptima posible, y con el uso de tecnologías que son utilizadas habitualmente en empresas de desarrollo de software como servicio para realizar la web.

Para ello, lo primero que hay que tener en cuenta es que las noticias deben de ser legibles por parte de todos los usuarios que accedan a la web independientemente de

si están registrados o no. Del mismo modo, todas las personas tienen derecho a expresar su opinión y, por lo tanto, si les interesa comentar alguna noticia deben de poder hacerlo. Si, al tiempo de ser lector de la web, un usuario quiere ver los artículos más recientes de un redactor en particular, debe de poder hacerlo. Obviamente, como todo servicio, nuestra web precisa de un mantenimiento, por lo que un usuario que esté registrado, y que sea administrador, debe poder acceder a unas pantallas que cualquier otro usuario, ya esté registrado o no y que no se corresponda con otro rol de administrador, no sea capaz de acceder para poder editar cualquier error que se pueda haber producido en el artículo o bien poder controlar los artículos que se muestran en la página principal.

Para el desarrollo del proyecto, he escrito un código en Java para gestionar una base de datos en la que se almacenan la información de las diferentes noticias y se puede acceder a ella para consultar estos datos por parte de la web. Por eso, he desarrollado un servicio que almacena y gestiona los artículos creados para poder mostrarlos mediante el desarrollo de la interfaz del usuario.

1.3. Estructura del trabajo

El resto de esta memoria se estructura de la siguiente manera:

- Capítulo 2. Antecedentes. En este apartado se realiza una breve introducción a determinados medios de comunicación e informativos que han sido criticados por, claramente, publicar noticias con un marcado carácter ideológico, frente a la neutralidad y el rigor periodístico que se debería mantener.

- Capítulo 3. Metodología de desarrollo. Este capítulo se centra en la descripción de las herramientas utilizadas, tanto en el desarrollo del Backend (Java, SQL) como en el Frontend (Thymeleaf) y despliegue de la aplicación (Docker)

- Capítulo 4. Diseño de la aplicación. Se desarrollará detenidamente el diseño de la aplicación web, indicando las diferencias que hay entre los diferentes

usuarios, y cómo se ha pensado que será el diseño de la base de datos de la aplicación.

- Capítulo 5. Implementación. En este capítulo hablaremos de los puntos más relevantes del desarrollo de la aplicación, explicando aspectos técnicos del código de la aplicación.

- Capítulo 6. Despliegue / Puesta en marcha. Finalmente, en este capítulo se explicará la manera de instalar y ejecutar en el equipo la aplicación, así como los prerrequisitos necesarios para ello.

Este trabajo concluye con el apartado de Discusión, exponiendo las conclusiones obtenidas del desarrollo de la aplicación, así como un breve apartado de futuros trabajos que podría servir para ampliar y mejorar la aplicación propuesta.

2. Antecedentes

Para poder saber en qué aspectos y vertientes me he inspirado para realizar el proyecto, he querido hacer un breve recorrido de la historia de los diarios en nuestro país, desde la primera publicación hasta llegar a las páginas digitales de los mismos periódicos o blogs de noticias. Así pues, a continuación menciono brevemente esta evolución hasta llegar a lo que me he basado para la realización del proyecto.

2.1. Primeras publicaciones físicas

Como bien sabemos, existen varios antecedentes históricos de periódicos comerciales, desde el periodico físico hasta llegar al periodico digital. Antes de este formato todos los periódicos eran de papel y se solían vender en los estancos cada mañana. Así pues, con la apertura de la Gaceta de Madrid en 1661, se da comienzo al periodismo, medio de comunicación que perdura hasta el día de hoy. La Gaceta de Madrid está considerada como el primer periódico semanal del país, siendo en la práctica una publicación similar a lo que hoy conocemos como Boletín Oficial del Estado (BOE).

A lo largo del siglo XIX fueron varios periódicos de renombre, algunos de los cuales siguen vigentes hoy, que dieron luz a este nuevo medio y teniendo un papel fundamental en la historia más reciente, dado que la prensa es la historia de la información. Algunos de los periódicos que se fundaron en aquella época y que a día de hoy siguen teniendo publicaciones son:

- **El Norte de Castilla**: fue el primer periódico de Castilla y León y se publica en Valladolid. Fundado en 1854 sigue teniendo publicaciones diarias abarcando a un público de más de catorce mil personas.
- **Las Provincias**: periódico editado en la Comunidad Valenciana y fundado en 1866. Cuenta con versión digital desde 1999, teniendo una tirada de más de dieciséis mil ejemplares.
- **La Vanguardia**: fue fundado el 1 de febrero de 1881, teniendo su sede principal en Barcelona, desde el 2011 se publica tanto en catalan como en castellano. A

día de hoy está considerado como uno de los cinco periódicos con más tirada y siendo el que más lectores tiene de todo el país.

- **La Voz de Galicia:** fundado en 1882. Es el diario gallego con mayor difusión y audiencia. Desde el 2000 cuenta con una edición digital siendo su página web una de las más visitadas de toda Galicia.
- **La Rioja:** fue fundado en 1889, se edita en Logroño y acapara más del 50% de ventas de periódicos en la comunidad riojana.

2.2. Primeras publicaciones digitales

A finales del siglo XX e inicios del XXI, se da la revolución digital en nuestro país, dando comienzo a la era digital en la cual estamos inmersos a día de hoy. Con esta revolución, los periódicos tuvieron que renovarse y apostar por la versión online, siendo complementaria, en un principio, de su versión física. Así pues, las primeras publicaciones digitales en nuestro país datan de finales de los noventa, siendo el BOE en 1994 quien dio el salto digital.

En 1994, la primera revista o periódico digital de carácter comercial fue una revista valenciana, *El Temps*, que inaugura su versión electrónica, adelantándose a los grandes periódicos de aquel momento. Poco a poco, los grandes periódicos fueron atreviéndose a realizar su versión online como puede ser la Vanguradia, eldiario.es o la Razón, entre otros.

2.3. Blogs y otras opciones digitales

Debido al gran crecimiento del mundo digital a partir del siglo XXI, donde mucha gente empezaba a tener sus propios ordenadores en casa y acceso a internet diario, se crean plataformas, blogs e, incluso, alguna red social donde compartir la información dando acceso a toda la sociedad. En este momento evolutivo entraron en juego las bien conocidas *fake news*, dado que todo el mundo tiene acceso a cualquier plataforma pudiendo editar o publicar lo que desea de manera anónima.

Así pues, las siguientes páginas me han servido de inspiración para poder llevar a cabo mi proyecto:

- **WikiTribune:** Es una plataforma para combatir contenidos falsos en la red con la publicación de noticias de calidad elaboradas de forma conjunta entre periodistas e internautas.
- **malagaldia.es:** Periódico social y colaborativo de Málaga y provincia, donde se quiere facilitar el éxito de los usuarios, anunciantes y colaboradores a través de diferentes formas de trabajo.

leonsurdigital.com: Magazín digital dedicado a noticias del Sur de León. Además de publicar las noticias en su web, se les puede encontrar en Facebook, Twitter e Instagram.

3. Metodología

Para realizar el proyecto, he decidido comenzar estableciendo, al igual que en las metodologías ágiles de desarrollo, los requerimientos de la aplicación para, con ello, poder definir cómo será la estructura de la aplicación y plantear así el flujo de trabajo necesario para que todo esté hecho en tiempo y forma.

El desarrollo se basa en la programación por capas, de modo que la aplicación consta de tres capas: la capa de datos, la capa de negocio y la capa de presentación. Así pues, el proyecto consta de tres grandes bloques: una base de datos para el almacenamiento de información, una aplicación en Java para desarrollar el Backend y una interfaz gráfica de usuario para desarrollar el Frontend.

Las siguientes secciones introducen los conceptos básicos y técnicos para el desarrollo de este trabajo. Comenzaré exponiendo la problemática de los tres grandes bloques, no sólo desde un marco más teórico, que se presenta en la sección 3.1, como puede ser la elección del tipo de base de datos, sino también desde el punto de vista práctico, como es la elección adecuada del software de desarrollo y las librerías necesarias, que se presentan en la sección 3.2.

Por último, en la sección 3.3 se presenta la planificación utilizada para el desarrollo del trabajo, tanto la estimada a priori, como la final.

3.1. Estructura de la aplicación

Como se ha explicado anteriormente, esta aplicación consta de tres grandes bloques: Base de Datos, Backend y Frontend. De cara a afrontar el desarrollo de la aplicación, se va a introducir cada uno de ellos por separado.

3.1.1. Base de datos

Una base de datos es un conjunto de datos almacenados y organizados que son accesibles desde alguna aplicación. Existen varios tipos de modelos de bases de datos entre las cuales cabe destacar las bases de datos jerárquicas, las transaccionales y, las más usadas hoy en día, las relacionales:

- Jerárquicas: Como el propio nombre indica, se almacenan los datos de forma jerárquica y son útiles para aplicaciones que requieren de un volumen grande de datos que son compartidos. La parte negativa de este modelo es la imposibilidad de representar los datos redundantes de una manera eficiente.

- Transaccionales: Su único fin es ser usadas para el intercambio de datos a grandes velocidades, como podría ser el intercambio de dinero.

- Relacionales: Como he mencionado antes, es el modelo más usado actualmente para administrar los datos dinámicamente y representar los problemas reales dado que el cómo y el dónde del almacenamiento de los datos no tiene relevancia, haciendo que sea más fácil de entender para cualquier usuario. Para acceder a la información se construyen unas consultas en lenguaje SQL (del inglés, *Structured Query Language*).

Por lo tanto, he decidido que el modelo de la base de datos será una base de datos relacional, por lo que es preciso encontrar un sistema de gestión de bases de datos relacional, como lo es PostgreSQL, como se expondrá más adelante.

3.1.2. Backend

El Backend es una capa de acceso a lo que ocurre entre un servidor y en una base de datos. Es la llamada programación en el servidor y también existe la maquinaria que funciona detrás de las escenas. No está accesible para el usuario final de la aplicación. Para implementar el Backend de la aplicación, he desarrollado una API propia para la aplicación.

Una API es el conjunto de funciones, procedimientos y rutinas que se utiliza para otra aplicación para así poder interaccionar con el programa. Las APIs intercambian órdenes y datos, lo que requiere protocolos y arquitecturas claras. Existen tres categorías de protocolos o arquitecturas de API: REST, RPC y SOAP. Cada uno de ellos puede llamarse "formato", con características y compensaciones únicas, y se emplea para fines distintos. En este proyecto he utilizado el protocolo REST.

REST se basa en un enfoque cliente/servidor que separa los extremos delantero y trasero de la API, proporcionando una flexibilidad considerable en el desarrollo y la implementación. REST es "sin estado", lo que significa que la API no almacena datos ni estado entre las solicitudes. REST admite el almacenamiento en la memoria caché, que guarda las respuestas para las API lentas o no sensibles a la vez. Las APIs REST, normalmente llamadas "APIs RESTful", también pueden comunicarse directamente u operar a través de sistemas intermedios como pasarelas de API. El cliente de una REST API puede ser un navegador web, un televisor o una aplicación Android, entre otras cosas. Toda la documentación sobre esta API puede encontrarse online en la siguiente url: getpostman API documentation[1].

Así pues, la parte Backend del proyecto consta de:

- Modelo: Contendrá mecanismos para acceder a la información y también para actualizar su estado. Los datos los tendremos en una base de datos, por lo que en los modelos tendremos todas las funciones que accederán a las tablas.

- API: Define las diferentes solicitudes que se pueden realizar desde el cliente y delega al controlador para realizar cada función.

- Controlador: Contiene el código necesario para responder a las acciones que se solicitan en la aplicación, como visualizar una noticia, crear un nuevo artículo, identificarte como usuario ya registrado, etc.

Con esta estructura la aplicación en el Backend funciona tal que la API recibe una petición por parte del cliente, para con las pantallas a mostrar, y la API le envía el listado de las pantallas a través del controlador que gestiona la información que es realmente útil de mostrar. Puesto que uno de los objetivos principales del proyecto es aprender a utilizar tecnologías que se usan en empresas que venden su software como servicio, para desarrollar el Backend he decidido utilizar el framework Spring.

Spring proporciona soporte de infraestructura como elemento clave en la capa de aplicación, proporcionando un modelo completo para la configuración y programación

[1] https://documenter.getpostman.com/view/11036932/Uz5Njt9U

de aplicaciones empresariales desarrolladas bajo Java, sin diferencia en el despliegue de la plataforma. Entre las características de Spring se ofrecen diversos servicios, entre los que se pueden destacar diversas tecnologías, como el enlace y acceso a datos, con soporte JDBC, Spring security y, la más importante, la inyección de dependencias. Las clases de una aplicación Java compleja deben ser lo más independientes posible de otras clases para así aumentar las posibilidades de ser utilizadas. La inyección de dependencias ayuda a unir estas clases y mantenerlas al mismo tiempo.

Entre las ventajas utilizar Spring, se puede nombrar que:

- Spring está organizado de forma modular, lo que significa que a pesar de la cantidad de paquetes y clases que tiene, solo debemos ocuparnos de aquellos que necesitemos para nuestro desarrollo e ignorar el resto.
- El framework web de Spring es un framework MVC (Modelo Vista Controlador) web bien diseñado.
- Probar una aplicación escrita con Spring es un proceso simple, porque el código dependiente del entorno se traslada a este framework.
- Cuenta con plantillas para diversas tecnologías entre la cuales podemos destacar las siguientes: JDBC, Hibernate y JPA, de forma tal que no hay necesidad de escribir un código extenso, ya que con estas plantillas simplifica el trabajo en cuanto a los pasos básicos a implementar de estas tecnologías.

3.1.3. Frontend

Cuando hablamos de Frontend nos referimos a la interfaz gráfica de usuario, la parte de la aplicación que ve el usuario final. Se utiliza un conjunto de elementos gráficos como imágenes, tablas, botones o formularios que facilitan la interacción del usuario con la aplicación. En esta interfaz se muestran los datos guardados en la base de datos y se gestionan los diferentes eventos para que el usuario pueda utilizar lo implementado.

Una de las cosas más importantes de esta parte, puesto que es un desarrollo para el usuario, es que sea usable y que la experiencia del usuario sea positiva. Para ello, es preciso que se cumplan varios principios importantes:

- Debe ser fácil de aprender. La interfaz debe ser fácil de aprender por los nuevos usuarios y fácil de recordar por los usuarios ocasionales.

- Debe ser visible. Es importante que los posibles cambios de estado del sistema sean visibles para el usuario. Por ejemplo, si estamos encima de un botón que el ratón pase a ser una mano o que el fondo del botón cambie de color.

- Debe ser eficiente. Una vez aprendido, la interfaz no debe ser un problema para la comunicación entre el usuario y el sistema, los tiempos de respuesta no deben superar los tiempos de reacción del usuario.

- Debe ser estéticamente agradable. La aplicación debe estar bien estructurada, no puede tener una colocación aleatoria que no sea entendible y debe tener diferentes características (diferentes formatos de la fuente, imágenes de un tamaño adecuado, etc) para que la experiencia del usuario sea agradable.

Por todo ello, el desarrollo web Frontend de la aplicación se ha creado usando los lenguajes HTML y CSS y un conjunto de herramientas para diseño de aplicaciones web llamado Bootstrap.

La elección de las herramientas, y el por qué de su elección en este trabajo, será explicado con más detalle en la siguiente sección.

3.2. Tecnologías usadas

Como se ha expuesto en la sección anterior, para cada parte de la aplicación se decidió cuál era el enfoque más adecuado a la hora de desarrollar el trabajo. Llegados a este punto se tuvo que hacer la elección del software y las librerías a utilizar para llevar a cabo este trabajo.

En ese apartado se explicará, de una manera más detallada, las diferentes tecnologías utilizadas para la creación de la aplicación.

3.2.1. PostgreSQL[2]

PostgreSQL es un sistema de gestión de bases de datos relacionales gratuito y de código abierto que hace hincapié en la extensibilidad y el cumplimiento de SQL. Cuenta con transacciones, vistas actualizables automáticamente, vistas materializadas, triggers, claves foráneas y procedimientos almacenados.

Algunas de las características más destacables de PostgreSQL son:

- Alta concurrencia. Es capaz de atender a muchos clientes al mismo tiempo.
- Soporte a triggers. Se pueden definir eventos y generar acciones cuando se disparan.
- Trabajo con vistas. Se pueden consultar los datos de manera diferente a como se guardan.
- Soporte para los lenguajes más usados. Se puede trabajar con funciones internas escritas en los lenguajes más comúnmente usados como C, C++, Java o Python.

He decidido utilizar PostgreSQL debido a que tengo un alto conocimiento de su funcionamiento, tanto en la creación de tablas o unique keys como en la manera de realizar las consultas, puesto que es mi día a día laboral.

3.2.2. IntelliJ Idea[3]

IntelliJ IDEA es un IDE desarrollado por JetBrains escrito en Java para desarrollar software informático con soporte para Linux, Windows y MacOS. Entre las funciones más importantes que me han hecho elegir este IDE podemos encontrar:

[2] https://www.postgresql.org/
[3] https://www.jetbrains.com/es-es/idea/

- Ayuda a la programación. Ofrece ciertas funciones autocompletar código mediante el análisis del contexto, la navegación por el código que permite saltar a una clase o declaración en el código directamente.

- Herramientas incorporadas e integración. Proporciona integración con herramientas de construcción/empaquetado como Gradle y Maven, soporta sistemas de control de versiones como Git y Mercurial y se puede acceder a bases de datos como PostgreSQL, SQLite y MySQL directamente desde el IDE.

- Soporte de plugins. Soporta plugins a través de los cuales se puede añadir funcionalidad adicional al IDE.

3.2.3. Spring Boot[4]

Spring Boot es una extensión de Spring para crear aplicaciones independientes basadas en Spring de nivel de producción que puede simplemente ejecutar. Está preconfigurada con la mejor configuración y uso de la plataforma Spring y de las bibliotecas de terceros. La mayoría de las aplicaciones de Spring Boot necesitan muy poca configuración de Spring. La elección de usar Spring Boot es debido a que es el framework de Java más usado mundialmente y, por lo tanto, uno de los más usados en las diferentes empresas que trabajan en Java. Podemos destacar algunas características como:

- Proporcionar POMs para simplificar la configuración de Maven

- Configurar automáticamente Spring siempre que sea posible
- Proporcionar características listas para la producción, como métricas, comprobaciones de estado y configuración externalizada
- No genera ningún código
- No requiere configuración XML.
- Integración compatible con todos los patrones de integración empresarial.

[4]https://spring.io/projects/spring-boot

3.2.4. BCrypt[5]

Como la protección de los datos de los usuarios registrados es algo esencial para cualquier Software, he decidido usar esta tecnología. BCrypt es un paquete al que podemos acceder gracias a la dependencia de *springframework security* y su función es implementar el hash de contraseñas, es decir, dada una contraseña cualquiera, la codifica de tal manera en caso de que se consiga cualquier intento de penetración en el sistema para robar las claves de acceso, dificulte la desencriptación de la contraseña. La salida final de la función BCrypt es una cadena de la forma:

$2a$10$j2fDDyHsKdXROl1wsKDmueGYF2qOlUHgO5basYb8iu8H3Odpj5itC

Donde:

- $2a$ es el identificador del algoritmo hash, que puede ser $2<a/b/x/y>$.
- 10 es el coste, el algoritmo usado por BCrypt es utilizado 2^10 veces.
- j2fDDyHsKdXROl1wsKDmue, es un hash aleatorio de 16 bytes (128 bits), codificado en radix-64 que tiene 22 caracteres.
- GYF2qOlUHgO5basYb8iu8H3Odpj5itC, es un hash aleatorio de 24 bytes (192 bits), codificado en radix-64 que tiene 31 caracteres.

3.2.5. Liquibase[6]

Liquibase es una biblioteca de código abierto e independiente de la base de datos para el seguimiento, la gestión y la aplicación de cambios en el esquema de la base de datos. Su función principal es facilitar el seguimiento de los cambios en las bases de datos. Los cambios se añaden en unos archivos XML, YAML o SQL y se identifican con la combinación de un identificador y el autor que ha realizado los cambios.

---

5 https://docs.spring.io/spring-security/site/docs/current/api/org/springframework/security/crypto/bcrypt/BCrypt.html

6 https://www.liquibase.org/

En estos archivos se pueden crear tablas, columnas (indicándose el tipo), relacionarlas con foreign keys e incluso añadir datos predefinidos a estas tablas creadas y es por esto último por lo cual he decidido usar esta tecnología, para que cualquier usuario que levante la aplicación tenga unos datos mínimos para poder apreciar el desarrollo.

3.2.6. Apache Maven[7]

Es una herramienta de gestión y comprensión de proyectos de software, la herramienta más usada para la construcción de proyectos Java, por lo cual ha sido elegido para el proyecto. Basada en el concepto POM (Project Object Model), puede gestionar la construcción, los informes y la documentación de un proyecto desde una pieza central de información. El objetivo principal de Maven es permitir la comprensión del estado del desarrollo en el menor tiempo posible. Para lograr este objetivo, Maven se ocupa de varias áreas de interés:

- Facilitar el proceso de construcción
- Proporcionar un sistema de construcción uniforme
- Proporcionar información de calidad sobre el proyecto
- Fomentar mejores prácticas de desarrollo

3.2.7. Thymeleaf[8]

Thymeleaf es un motor de plantillas Java XML/XHTML/HTML5 que puede funcionar tanto en entornos web como no web. El objetivo principal es aportar plantillas naturales al flujo de trabajo de desarrollo y es la opción más adecuada para usar HTML5 en la capa de vista de las aplicaciones web basadas en MVC Proporciona una completa integración con Spring Framework y es ideal para el desarrollo web moderno de HTML5 JVM. Por todo ello es por lo que ha sido elegida tecnología para ayudar al desarrollo del Frontend.

[7] https://maven.apache.org/
[8] https://www.thymeleaf.org/

3.2.8. Bootstrap[9]

Bootstrap es un framework de desarrollo Frontend de código abierto para la creación de sitios web y aplicaciones web. El marco de trabajo de Bootstrap se basa en HTML, CSS y JavaScript (JS) para facilitar el desarrollo de sitios y aplicaciones con capacidad de respuesta. El diseño responsivo permite que una página web o una aplicación detecte el tamaño y la orientación de la pantalla del usuario que accede a ella y adapta automáticamente la visualización.

Bootstrap es muy utilizado mundialmente, por lo que existe mucha documentación y muchos de los posibles problemas que me podría encontrar están resueltos por otros usuarios, lo cual me serviría mucho debido a que mi conocimiento respecto al desarrollo Frontend era muy escaso.

3.2.9. Docker[10]

Docker es una plataforma de código abierto para desarrollar, enviar y ejecutar aplicaciones que permite separar sus aplicaciones de la infraestructura para que se pueda entregar el software rápidamente. Con Docker, se puede gestionar la infraestructura de la misma manera que gestiona las aplicaciones.

Ofrece la posibilidad de empaquetar y ejecutar una aplicación en un entorno poco aislado llamado contenedor. El aislamiento y la seguridad le permiten ejecutar muchos contenedores simultáneamente en un determinado host. Los contenedores son ligeros y contienen lo necesario para ejecutar la aplicación, por lo que no es necesario depender de lo que está instalado actualmente en el host.

Al aislar la aplicación en contenedores, nos podemos asegurar que la aplicación funcionará en cualquier equipo que tenga Docker instalado sin depender de su sistema operativo. Es por eso que he decidido usar esta tecnología indicando en el proyecto la manera de iniciarlo.

[9] https://getbootstrap.com/
[10] https://www.docker.com/

3.3. Planificación del trabajo

Con el planteamiento de la estructura de la aplicación realizado, así como la elección de las tecnologías a utilizar, es preciso organizar las etapas de desarrollo del trabajo con el fin de preparar una programación adecuada de las tareas que se van a realizar.

Al iniciar el trabajo, se contaba con 15 semanas para realizar todo el proyecto, reservando las últimas 4 semanas para la redacción de la memoria. Con lo cual el trabajo de desarrollo debería ser realizado en sólo 11 semanas.

Marqué 3 semanas para realizar la parte Backend y Base de datos del proyecto, 4 semanas para la parte Frontend, 2 para la puesta en producción y 2 para realizar pruebas con usuarios reales. De un modo más detallado, la planificación está expuesta en la Figura 1, mediante un diagrama de Gantt. mostrando los plazos de cada tarea.

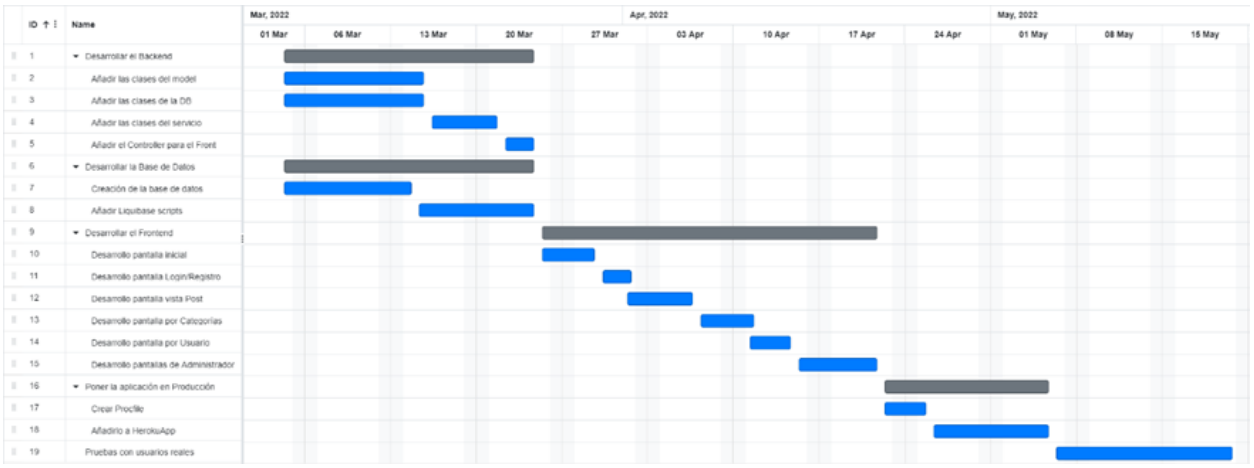

Figura 1. Planificación. Diagrama de Gantt desarrollado al comienzo de este proyecto, mostrando los plazos previstos para el desarrollo de cada tarea.

Como se puede ver a continuación en el diagrama final, Figura 2, donde se muestran los tiempos reales de ejecución durante el desarrollo del trabajo, el planteamiento inicial subestima muchas de las tareas de Backend, puesto que en principio iba a ser una parte más fácil, debido a que tengo un mayor conocimiento en este campo, frente al desarrollo de Frontend, que debería llevarnos más tiempo de desarrollo. Por ello, tuve que trabajar con el controlador del Frontend a la vez que diseñaba las pantallas,

retrasando los tiempos esperados al inicio de este proyecto. Además, me encontré con un problema ajeno al planteamiento inicial a la hora de hacer la puesta en producción por el que tuve que cambiar la aplicación usada y apurar hasta el final las pruebas con los usuarios.

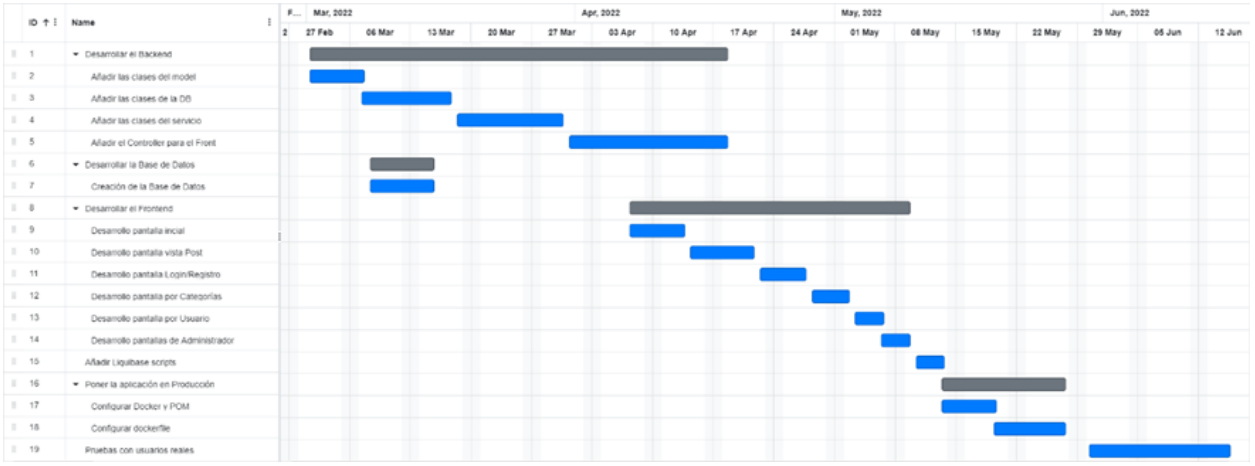

Figura 2. Diagrama de Gantt que muestra el desarrollo del proyecto, con los tiempos reales de desarrollo del trabajo.

4. Diseño

Una vez hemos definido los requisitos de la aplicación, tenemos la idea de lo que queremos hacer y se ha analizado lo que se quiere para la aplicación, procedemos a hacer el diseño de la aplicación. Es importante que el diseño que se realice de la aplicación se anticipe a lo que se pueda requerir en el futuro. Para ello, empezaremos realizando el modelo de la Base de Datos.

4.1. Base de datos

Para la aplicación propuesta, la base de datos es esencial, de modo que los usuarios se puedan registrar y, a su vez, se puedan almacenar y gestionar las noticias. Para ello se puede desarrollar un modelo simple pero que tenga todo lo necesario, implementando una serie de tablas que podemos catalogar como tablas de usuario, tablas base y tablas relacionales.

4.1.1. Tablas de usuario

Las tablas de usuario nos permiten guardar la información de los usuarios registrados que tendrán capacidad de añadir comentarios o gestionar la aplicación, dependiendo de sus permisos.

Estas tablas son: *user_table*, *role* y *permission*.

- ➔ ***User_table***. Es la tabla que contiene los diferentes usuarios registrados y la información personal de cada uno de ellos, como por ejemplo su nombre de usuario (*username)* y la correspondiente contraseña, necesarios para identificarse, o el email en caso de que se le deba notificar algo. También identifica al usuario con un rol. Como se ha comentado, la aplicación puede ser usada sin la necesidad de estar logueado, pero para usar algunas funcionalidades es necesario estar identificado. La finalidad de muchos diarios es guardar esta información para venderla a terceros, pero en este caso es simplemente para mantener un control en caso de que la conducta de algún usuario no sea correcta. El nombre de esta tabla es *user_table* porque la palabra *user* está guardada por PostgreSQL y no se puede usar como nombre de tabla.

- ***Role***. Es la tabla que almacena los diferentes roles o conjunto de permisos que se asigna a cada usuario, según la función que desempeña. La única información necesaria de un rol es su nombre para que sea fácil de identificar y, por lo tanto, es el único campo que se guarda en esta tabla. En esta primera versión de la aplicación hay dos posibles roles a asignar: usuario externo y/o administrador.

- ***Permission.*** Es la tabla que almacena los diferentes permisos. Un permiso es un conjunto de entidad y operación en la que una operación puede ser de lectura, de creación de actualización o de borrado y una entidad es el objeto que quiere ser visto, modificado, etc. La información necesaria de un permiso es su nombre para que sea fácil de identificar, por lo que es el único campo que se guarda. En esta primera versión de la aplicación, el modelo de permisos no funciona así, puesto que solo hay dos tipos de roles y las pantallas de administrador se han podido ocultar sin la necesidad de entidades, pero puede ser usado en un futuro.

4.1.2. Tablas base

Estas tablas son las mínimas necesarias para que la aplicación tenga un sentido y pueda funcionar, guardando la información necesaria de un post. Gracias a estas tablas, se puede mostrar en el navegador lo necesario para que lo vea el usuario.

Se han definido cuatro tablas diferentes: *post*, *category*, *section* y *comment*.

- ***Post*.** Contiene los artículos que se han escrito. Entre la información destacada se guarda el título, el subtítulo o el mismo contenido del artículo. También se guarda el usuario que lo ha creado así como la categoría a la cual pertenece.

- ***Category*.** Almacena las diferentes categorías. La información necesaria de una categoría es su nombre para que sea fácil de identificar, por lo que es el único campo que se guarda. Las categorías se usan para indicar el tema del cual trata la noticia.

- ***Section***. Almacena las diferentes secciones. La información necesaria de una sección es su nombre para que sea fácil de identificar, por lo que es el único campo que se guarda. Una sección es una pequeña categorización de las noticias para ser mostradas en la pantalla inicial, lo que comúnmente se conoce como la portada. Las secciones corresponden con las categorías, pero tienen un comportamiento diferente. Las secciones se usarán en la portada mostrando como mucho 4 artículos por categoría además de los 5 últimos artículos creados que serán los que se mostrarán primero, independientemente de su categoría.

- ***Comment.*** Almacena los diferentes comentarios escritos por los usuarios en cada noticia. Entre la información destacada de un comentario está el contenido en sí, el nombre del usuario que ha escrito dicho comentario y la información del post que ha sido comentado. Es la parte social de la aplicación en la que todo el mundo puede decir lo que le apetezca, siempre manteniendo un buen clima de respeto hacia el resto de usuarios, que será controlado por los administradores y que podría llegar a implicar la pérdida de la cuenta.

4.1.3. Tablas relacionales

Las tablas relacionales son utilizadas para definir las relaciones entre las otras dos tablas ya existentes, indicando en cada fila de la tabla el identificador que se quiere relacionar.

Estas tablas son: *post_section* y *role_permission*.

- ***Post_section*.** Relaciona los artículos que se han escrito con las secciones en las que se debe mostrar en la pantalla principal. Puesto que la sección es una parte de la portada, es necesario que se indique qué artículos queremos que se muestren en las diferentes secciones.

- ***Role_permission*.** Relaciona los roles con los diferentes permisos. Como se ha indicado previamente, un rol es un conjunto de permisos y esta es la tabla en la que se relacionan los diferentes permisos en un rol único.

4.1.4. Diagramas

Para aclarar el diseño de la base de datos que integra la aplicación y mostrar las relaciones entre las diferentes tablas, a continuación, se muestran dos diagramas diferentes, que son los más importantes en el diseño de las bases de datos.

La primera imagen, Figura 3, corresponde con el diagrama Entidad-Relación de las tablas más importantes de nuestra base de datos, mientras que en la Figura 4 se muestra el modelo relacional de las tablas.

Un diagrama Entidad-Relación es una herramienta para modelar datos que permite representar a las entidades relevantes de un sistema de información y sus interrelaciones y propiedades. Se usa frecuentemente para diseñar una base de datos de manera conceptual en la fase de desarrollo de software.

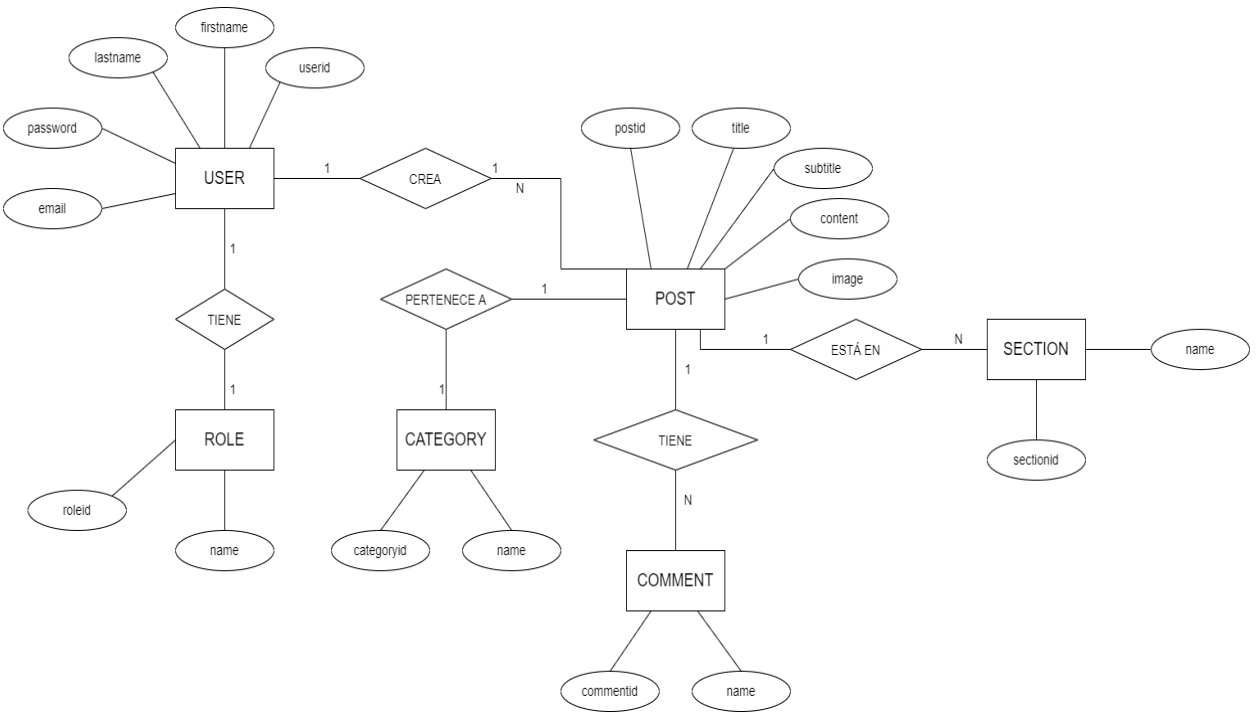

Figura 3. Diagrama Entidad-Relación de las bases de datos.

El modelo relacional considera la base de datos como un conjunto de relaciones. Una relación es un conjunto de columnas, también llamadas campos o atributos, y filas, también llamadas tuplas o registros, donde cada fila es un conjunto de campos, las columnas, que representan valores que describen el mundo real.

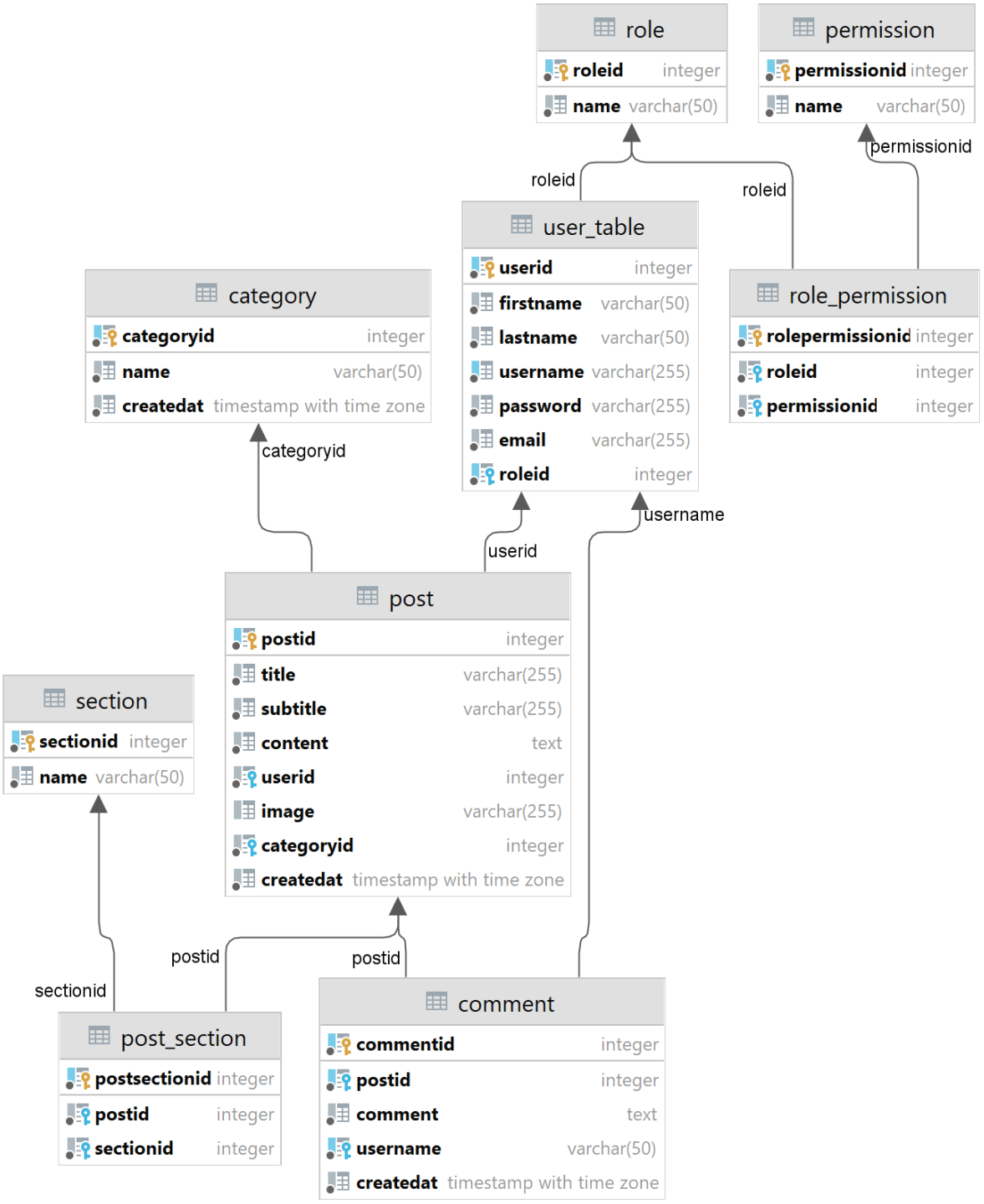

Figura 4. Modelo relacional de las tablas que componen la base de datos.

4.2. Aplicación web

Como hemos dicho, la aplicación tiene que ser accesible independientemente de si el usuario está identificado o no, por lo que el acceso será directo. Es decir, un usuario no registrado será capaz de acceder libremente a la información, aunque será necesario estar registrado para realizar determinadas acciones

En resumen, existen tres tipos de acceso en esta aplicación: acceso sin registro, acceso registrado como usuario externo y acceso registrado como administrador.

4.2.1. Acceso sin registro

Para el acceso sin registro, lo único que deberemos de hacer será acceder a través de un navegador web a nuestro localhost (http://localhost:8080) donde accederemos a la portada principal.

El diseño de las páginas será similar. Contarán con un encabezado en el cual, en la esquina superior izquierda nos encontraremos con el logo de la aplicación y en la derecha veremos, para un usuario que no esté identificado, dos botones diferentes: *Login*, para identificarse si ya ha creado el usuario con anterioridad y por lo tanto existe en la base de datos, y *Register*, para registrarse como nuevo usuario en la base de datos. Justo debajo del encabezado encontraremos las diferentes categorías a las cuales pertenecen los diferentes artículos.

El usuario podrá ver las diferentes noticias destacadas que se muestran en la portada, dividida en secciones, y podrá acceder a los artículos para leerlos en profundidad. También podrá acceder a diferentes pantallas que pueden ser de interés para cualquier usuario, puesto habrá varios botones como los que se pueden encontrar en cada artículo encima de la imagen, de color rojo, que corresponderá con la categoría del artículo en el que si clicamos nos dirigirá a una página en la que veremos los 10 últimos artículos creados que pertenezcan a esa categoría. Otro ejemplo de estos botones es el botón que hay en el nombre del usuario que ha escrito el artículo y que, si clicamos en él, nos dirigirá a otra pantalla en la que veremos todos los posts creados por ese usuario.

En la pantalla de registro, aparecerá un formulario en el que la persona que esté interesada en registrarse lo único que debe completar es su nombre, su apellido, un correo electrónico, que será usado como su nombre de usuario, y una contraseña. Un detalle importante es que por defecto cualquier usuario que se cree a través de la parte frontal de la aplicación será un usuario externo. Una vez el usuario está registrado, ya puede identificarse donde lo único que debe introducir es el correo introducido en el registro y la misma contraseña.

Con esto termina lo que podrá ver un usuario que no esté identificado como usuario de la aplicación, donde podrá acceder a todas las noticias para poder estar informado.

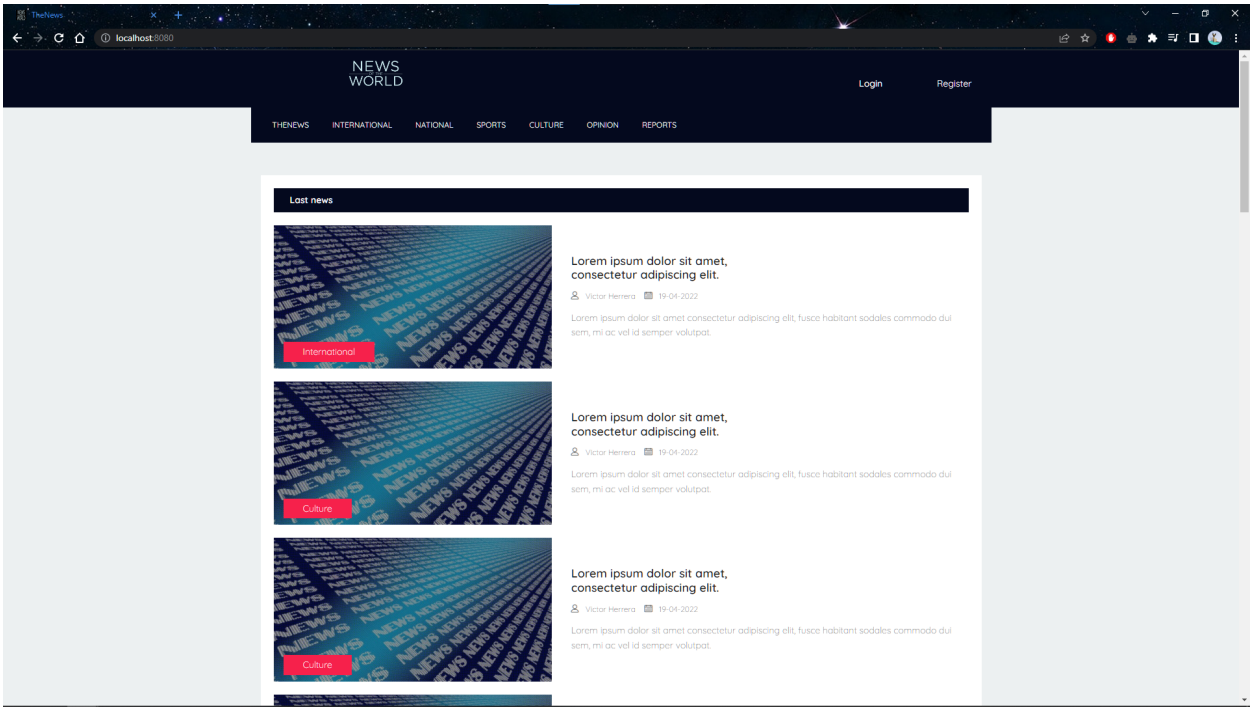

Figura 5. Vista de la portada sin identificación.

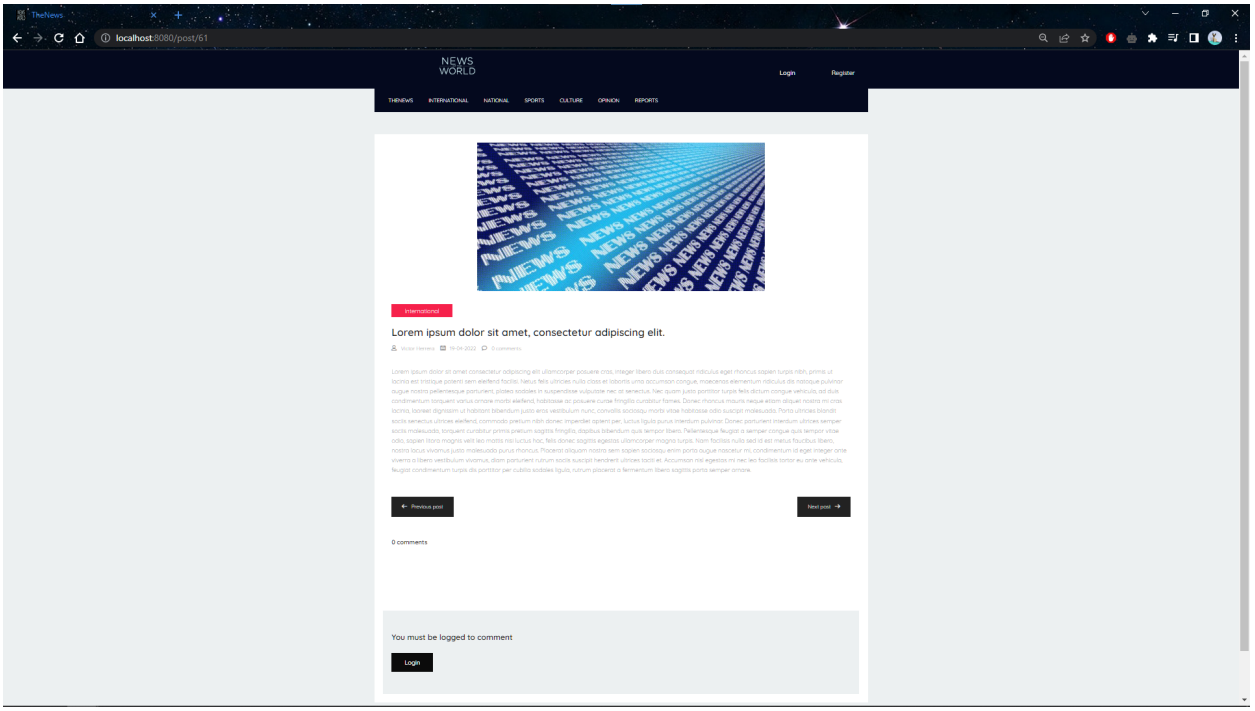

Figura 6. Vista de un artículo sin estar identificado en el que no podemos escribir un comentario.

4.2.2. Acceso como usuario externo

Para el acceso como usuario externo es necesario que previamente se haya creado un usuario a través de la pantalla. Deberemos de acceder a través de un navegador web a la dirección http://localhost:8080/login, donde accederemos a la pantalla de identificación en la que se debe de indicar el nombre de usuario, que corresponde con el mismo correo que se puso a la hora de registrarse, y la contraseña puesta en el registro.

Una vez identificado, el cabezal de la página mantendrá la misma estructura, donde en la esquina superior izquierda veremos el logo de la aplicación pero en la esquina derecha veremos ahora nuestro nombre de usuario y un botón para cerrar sesión y dejar de estar identificado y pasar a ver de nuevo todo como un usuario sin estar identificado. En la barra de categorías todo se verá igual que en el caso del usuario no identificado.

La otra diferencia respecto al usuario que no está identificado es que al acceder a un artículo tenemos permitido añadir comentarios sobre la noticia. Para añadir el

comentario es preciso que añadamos nuestro nombre de usuario y luego podremos añadir el texto que nos apetezca, siempre dentro de los límites del respeto.

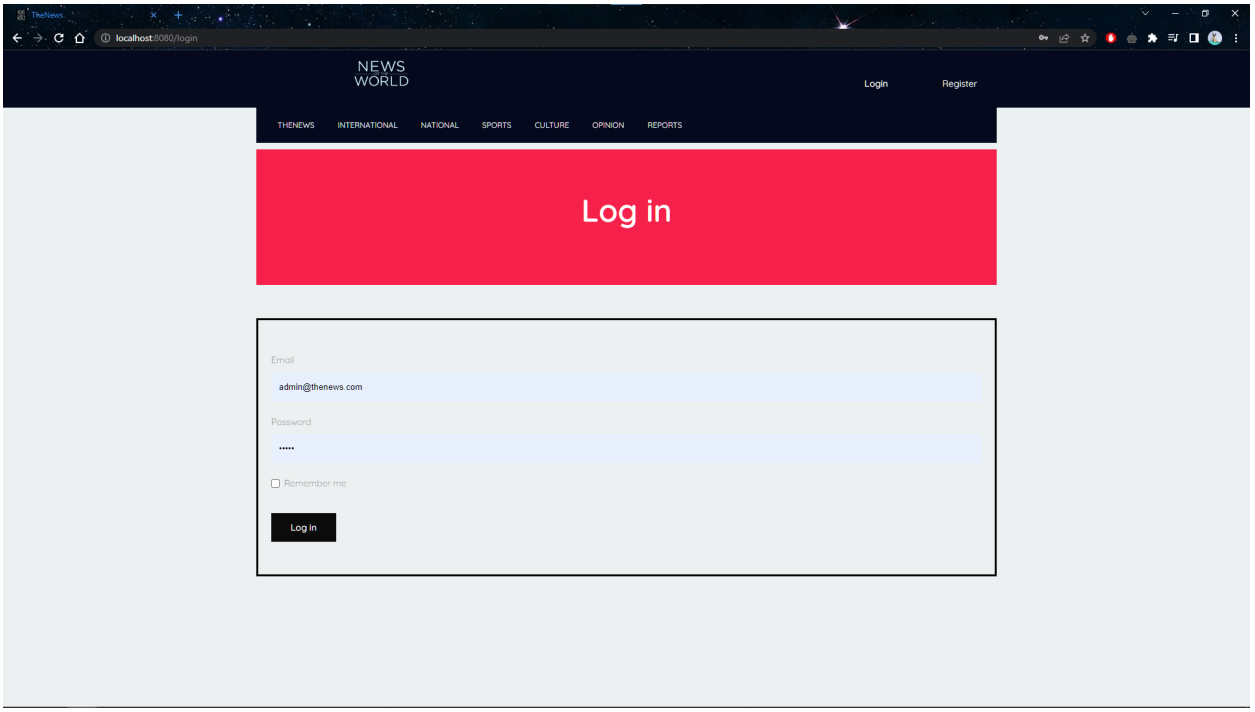

Figura 7. Vista de la pantalla de identificación.

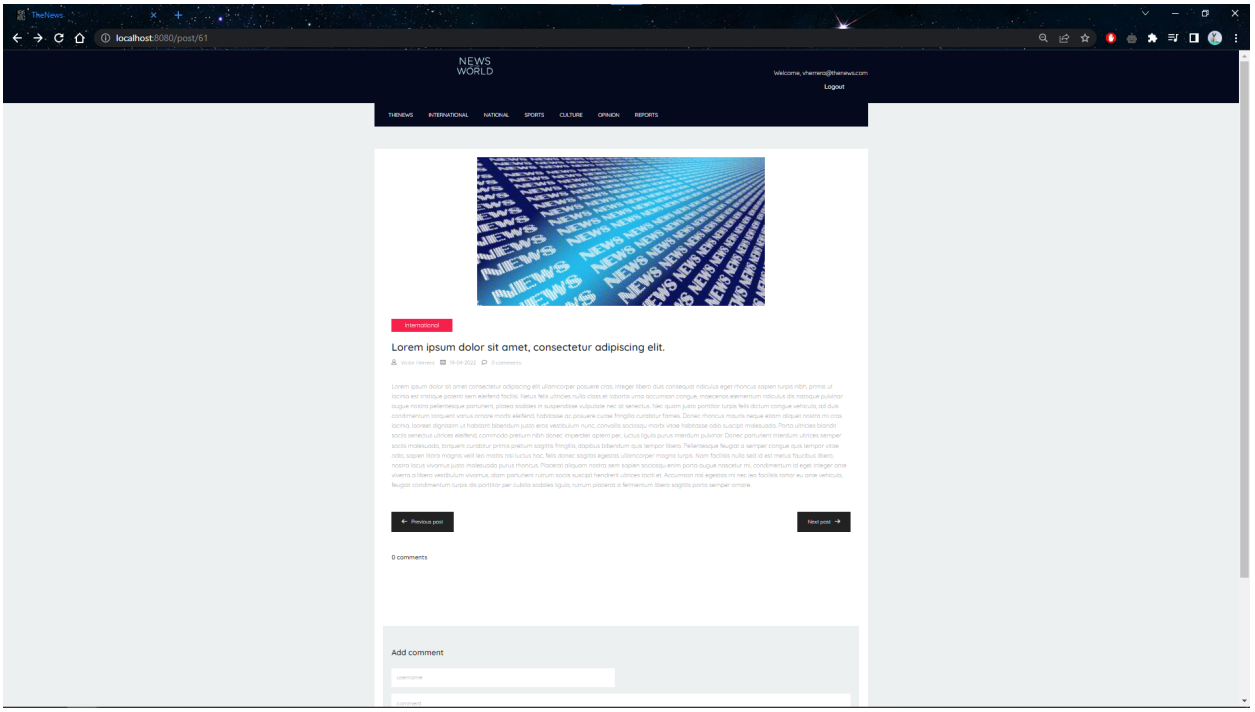

Figura 8. Vista de un artículo estando identificado en el que podemos escribir un comentario.

4.2.3. Acceso como administrador

Para el acceso como administrador es necesario que previamente se haya creado un usuario a través de la API, puesto que cualquier usuario creado por la pantalla será creado como usuario externo. Deberemos de acceder a través de un navegador web a la dirección http://localhost:8080/login donde accederemos a la pantalla de identificación en la que se debe de indicar el nombre de usuario y la contraseña de un usuario que tenga asignado el rol de administrador. Para la realización de las pruebas, se puede usar el usuario administrador ya creado cuyo nombre de usuario es *admin@thenews.com* y la contraseña es el nombre de usuario.

En el encabezado de la página, el usuario administrador verá lo mismo que ve el usuario externo, el logo a la izquierda y a la derecha su nombre de usuario y el botón para cerrar sesión. En la barra donde aparecen las categorías, verá otro menú donde podrá realizar las funciones propias del administrador de la página: añadir un nuevo artículo o gestionar los artículos.

En esta primera versión solo existirán dos roles para los usuarios registrados, el de administrador y el de usuario externo, por lo que el añadir nuevos artículos tendrá que ser hecho por el administrador de la página a pesar que el artículo no lo haya redactado él. En la pantalla de la creación de un artículo aparecerá un formulario en el que se deberá de rellenar el título, el subtítulo, la URL de la imagen y el contenido del artículo, así como elegir la categoría a la cual pertenece el artículo y el usuario que lo ha redactado, que puede ser distinto al que lo está creando.

En la pantalla de gestión de artículos aparecerán en una tabla todos los artículos que existen en la base de datos ordenados por fecha de creación descendente, los más nuevos aparecerán arriba. Además del título, la categoría y la fecha de creación del artículo, el usuario administrador podrá realizar tres acciones para cada artículo: editarlo, eliminarlo o editar la sección a la que pertenece.

- **Editar un artículo.** La pantalla de edición de un artículo tendrá la misma estructura que la pantalla de creación del artículo, pero se verán todos los campos rellenados con los valores que hay guardados en la base de datos. Se podrá editar un campo, todos o ninguno de ellos.

- **Eliminar un artículo.** Eliminar un artículo no tendrá una pantalla propia, solo será un botón. Una vez accionado el botón internamente se eliminará la relación entre el artículo y la sección y se eliminará el artículo de la base de datos.

- **Editar la sección del artículo,** En la pantalla de edición de la sección, aparecerá prácticamente la misma información que en la pantalla previa pero solo para el artículo que se va a editar. Ahora, en vez de aparecer la fecha de creación del artículo, aparecerá un desplegable con las diferentes secciones que existen en la portada. Por defecto se mostrará la sección que ya tiene asignada en la base de datos, pero si esta relación todavía no existe, se mostrará la primera sección según el orden del identificador en la base de datos.

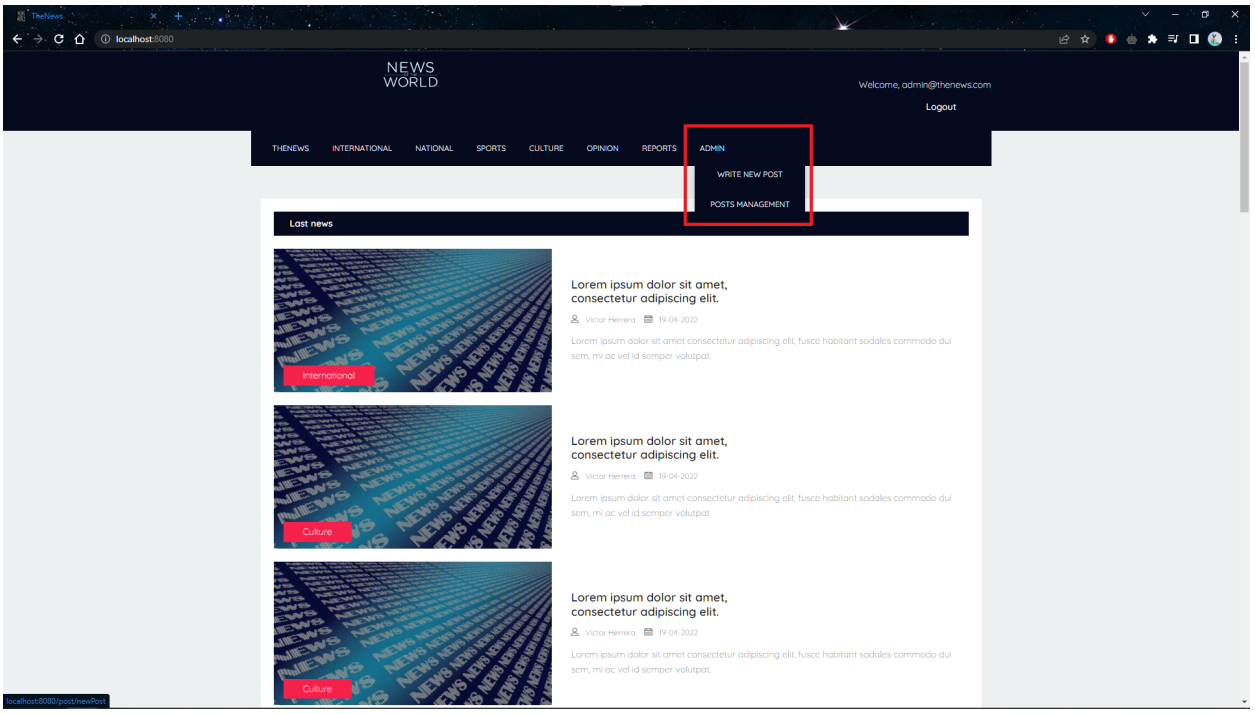

Figura 9. Vista de la portada estando identificado como administrador en el que vemos los nuevos menús.

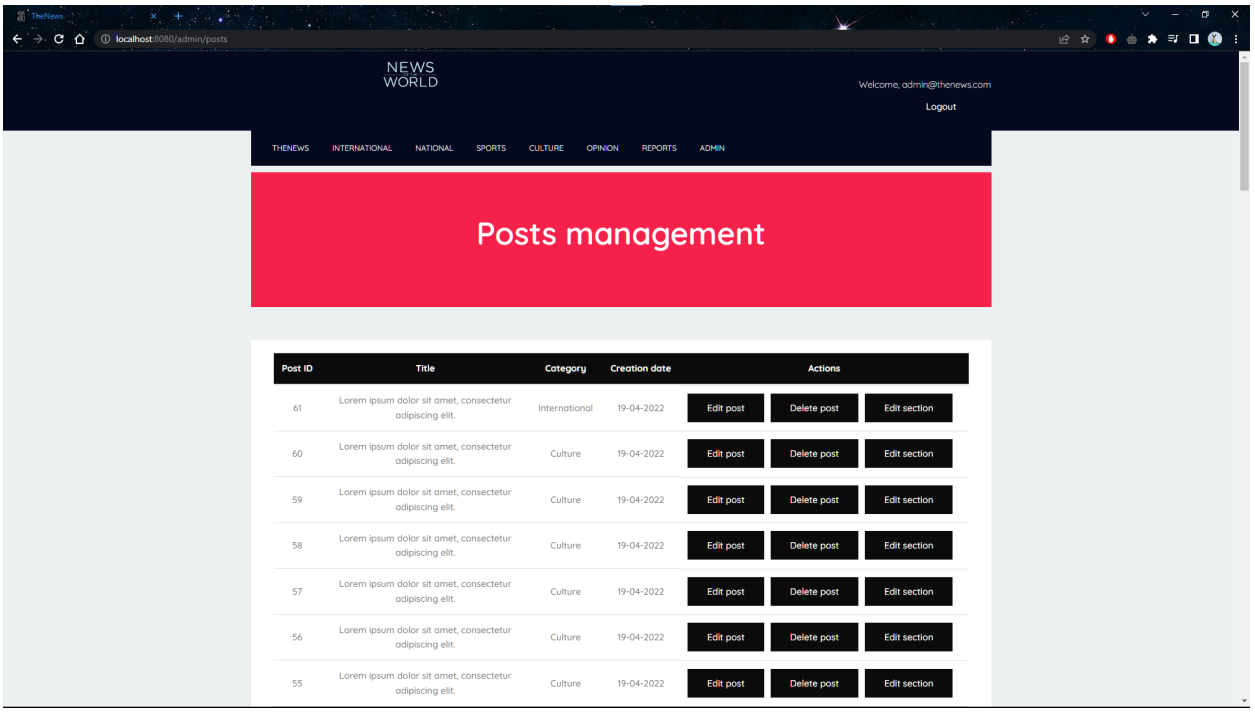

Figura 10. Vista de la pantalla de gestión de artículos.

5. Implementación

Como ya tenemos bien claro cuales son las tecnologías que vamos a usar y el diseño del proyecto, nos disponemos a programar la aplicación para que sea tal y como nos la hemos planteado.

La implementación de la aplicación se ha realizado en tres bloques diferenciados.

5.1. Base de datos

La implementación de la base de datos se ha realizado mediante la creación de archivos XML que, gracias a Liquibase, crean las tablas y las relaciones entre ellas, descritas en el punto 4.1. y que se puede ver reflejada en la Figura 4 del mismo apartado, así como el relleno de estas tablas con los datos mínimos para que la aplicación se pueda visualizar sin problemas.

A continuación se explicará brevemente la estructura de los archivos.

5.1.1. Master

El archivo *changelog-master.xml* es, como se puede intuir por su nombre, el archivo que va a ser la raíz de los cambios en la base de datos y que se tiene que ir actualizando con los diferentes cambios que se vayan a hacer en la base de datos en versiones futuras de la aplicación.

En este archivo lo que se incluye son los diferentes archivos con datos que tienen las diferentes versiones de las tablas de la base de datos y los datos para rellenar las tablas. En futuras versiones de la aplicación en la que potencialmente se usen más tablas o se necesiten más datos que sean básicos para la aplicación, lo único que habrá que hacer será crear los archivos con los cambios que se han hecho en las tablas o en los datos que hay en esta primera versión y ponerlos a continuación de este archivo master.

Para que la aplicación sepa que este es el archivo que debe revisar para el control de los cambios en la base de datos, se debe de incluiren el application.properties con la siguiente estructura:

➔ spring.liquibase.change-log=classpath:db/changelog/changelog-master.xml

5.1.2. Data

Como hemos comentado en el apartado anterior, la creación de la base de datos y el relleno de los datos se hace en unos archivos que se definen en el archivo master.

En la implementación de la aplicación he decidido crear un archivo que tenga todos los changeset que incluyen la creación de las tablas y columnas de la base de datos física, así como identificar las primary keys, las foreign keys o las unique keys. En este archivo se desarrolla la base de datos tal y como está en la Figura 4.

Tras crear la base de datos, se añaden roles, usuarios, artículos y el resto de la información para que al iniciar la aplicación por primera vez, la aplicación muestre datos de tal manera que la aplicación pueda ser observada y probada. Cabe destacar que en cada introducción de datos se hace una comprobación previa para comprobar si los datos ya existen, en base a la unique key que se ha configurado previamente y, en caso de que ya exista, no se hace ninguna inserción en la base de datos.

5.2. Backend

Con la base de datos creada, toca implementar la API de la aplicación. En este punto vamos a explicar la organización del código de la aplicación, agrupado por paquetes, y cómo interaccionan entre ellos.

5.2.1. Model

En este paquete encontramos las clases que son el modelo de la aplicación. Encontramos 9 clases que identifican las diferentes tablas de la base de datos y en cada una podemos encontrar los atributos de ellas, las columnas. Dentro de estas

clases, encontramos los Getters and Setters que serán utilizados en el desarrollo de otras partes del Backend de la aplicación.

Además, encontraremos una clase más, llamada BaseDBModel, que se utilizará en las clases del controlador de la base de datos para poder hacer edición de cualquier tipo en las tablas de la base de datos a través de diferentes métodos.

5.2.2. Database

En el paquete Database veremos 10 interfaces y un subpaquete llamado *impl* con 9 clases. Esta estructura está hecha pensando en un futuro puesto que reúne dos ventajas de la herencia: favorecer el mantenimiento y la extensión de las aplicaciones ya que se permite la existencia de variables y métodos polimórficos. De la misma manera, hace que para llegar al código final, la implementación real de los métodos, no sea tan directa.

Entre las 10 interfaces nos encontramos las 9 interfaces de las clases de modelo, así como la interfaz que sirve de padre de las demás interfaces que se implementan posteriormente en las que están los métodos que se consideran base de todas ellas, puesto que son las operaciones básicas que se puede hacer en una base de datos: consultar filtrando por identificador, mostrando la lista completa, creando una fila, actualizando un registro o eliminandolo.

Por lo tanto, en las clases, además de implementar los métodos propios que cada interfaz requiera, también encontramos la implementación de los métodos que son base. En estos métodos lo que se hace es construir una query de PostgreSQL para ser ejecutada y devolver lo que se requiera en cada método.

5.2.3. Mapper

En el punto anterior hemos mencionado la creación de métodos en los que se podían consultar algunos registros de la base de datos o incluso una tabla entera. Pero lo que nos devuelve la base de datos después de haber realizado la consulta no lo devuelve en código Java. Es por eso que necesitamos de las clases que nos encontramos en este paquete.

Una clase mapper es un traductor entre lo que la base de datos devuelve como respuesta y el atributo que es según el modelo de la API. Así pues, hay una clase mapper para cada clase modelo que implementa un método llamado *mapRow* de la interfaz *RowMapper<T>* para asignar cada fila de datos con un objeto del tipo modelo. Para hacerlo, se crea un nuevo objeto y se usan los *Setters* definidos en el modelo.

5.2.4. Service

Las clases de servicio son aquellas que implementan los métodos que son utilizados a posteriori por el controlador del Frontend. Es por ello que, en este caso, no hay una clase para cada modelo, puesto que, por ejemplo, todo lo que es referente a los roles y permisos no va a dar información útil al Frontend, ya que esto se gestiona desde la parte de seguridad que veremos en el punto 5.2.6.

Así pues, la estructura del paquete de servicio tiene 7 interfaces y un subpaquete donde se implementan. Los métodos de la implementación son muy simples, puesto que esta es la capa de negocio y todo lo que es el código está ya desarrollado en la parte de la database.

Hay una pequeña excepción a lo expuesto en el anterior párrafo, y es que una de estas interfaces e implementaciones no corresponde con uno de los modelos implementados, sino que es un modelo para la portada de la aplicación, aunque será usado también para otras pantallas.

5.2.5. Controller

El paquete del Controller tiene dos bloques diferenciados, los controladores de la base de datos y los controladores del Frontend que se encuentran en un subpaquete llamado *mvc*.

Las clases controlador de la base de datos son la construcción de la API REST. En cada clase encontramos los diferentes métodos que se han implementado para modificar la base de datos siendo llamados por una de las siguientes funciones de

REST: GET, POST, PUT y DELETE. Estos métodos pueden ser usados en una plataforma de gestión de APIs como puede ser Postman[11].

En el subpaquete *mvc* nos encontramos con 5 clases en las que se vinculan los métodos creados en el servicio con los HTML donde se definen la estructura de las diferentes pantallas, podría decirse que son unos conectores. En ellas hay diferentes métodos en los que se envía la información necesaria para gestionar lo que aparece en pantalla y las diferentes acciones que se puedan hacer en ellas.

5.2.6. Security

Puesto que queremos ocultar algunas partes de las pantallas según el rol del usuario, así como hacer que el usuario pueda identificarse sin vulnerar su usuario y tratando de mantener su información privada segura, hemos añadido Spring Security[12].

Por eso, tenemos el paquete de security en el que hay un modelo y un servicio para los detalles de seguridad de un usuario y una configuración hecha para que lo descrito anteriormente de ocultar algunas partes o no permitir algunas acciones pueda llevarse a cabo.

En la configuración de la seguridad, lo más destacable es el método *configure* en el cual se especifica que pantallas pueden ser vistas o no dependiendo del rol asignado al usuario, así como la pantalla en la cual el usuario puede identificarse y así poder disfrutar de toda la aplicación.

[11] https://www.postman.com/
[12] https://spring.io/projects/spring-security

5.2.7. Diagramas UML

Para aclarar la implementación del Backend y la interacción de los diferentes paquetes que integran la aplicación, a continuación, se muestran dos diagramas de la interacción de dos modelos.

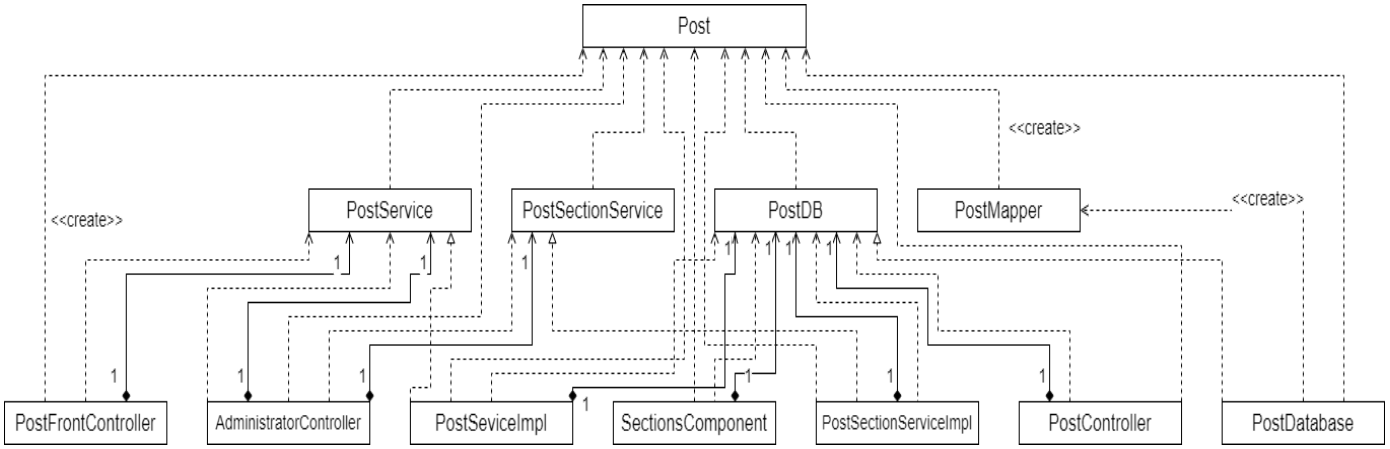

Figura 11. Plantilla UML de la interacción del modelo Post con el resto de clases e interfaces.

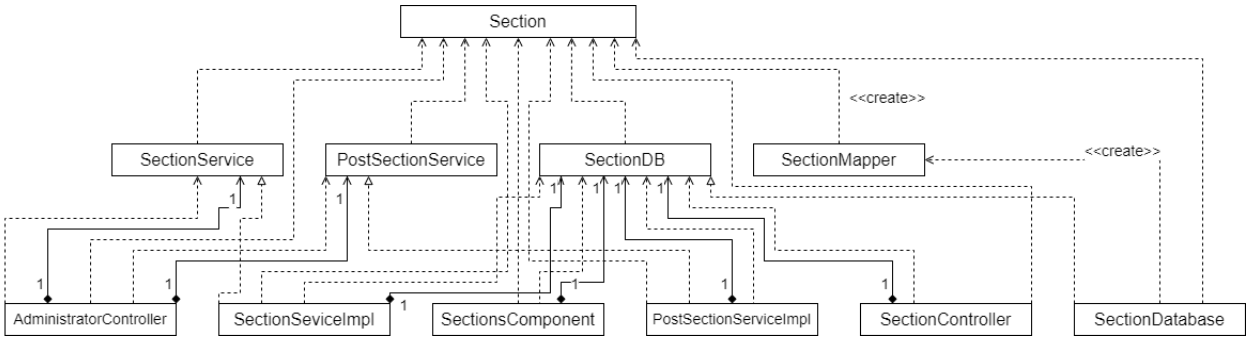

Figura 12. Plantilla UML de la interacción del modelo Section con el resto de clases e interfaces.

5.3. Frontend

El diseño del Frontend es tal y como se planteó en el diseño así que en esta sección no entraremos mucho en detalle de esto y si en algunos aspectos técnicos de la implementación.

5.3.1. Thymeleaf

Como se ha comentado en el punto 1.1, una de las motivaciones para hacer este trabajo es el aprendizaje de nuevas tecnologías. Por eso, para mostrar por pantalla la información que está guardada en base de datos y que a través del controlador nos llega gracias al servicio, he decido usar Thymeleaf.

El uso de Thymeleaf va integrado en el mismo código HTML y nos ayuda a que, en el caso que el método en cuestión nos devuelva más de un objeto a mostrar, no debamos duplicar el código por cada elemento y solo preocuparnos de que el desarrollo funcione al menos para uno. También lo hemos utilizado para mostrar los diferentes títulos, subtítulos, categorías, etc. de cada artículo, puesto que se puede indicar el valor para cada objeto dentro de una iteración, como un bucle *for* de Java.

Además, Thymeleaf tiene un paquete extra en el cual podemos encontrar Thymeleaf SpringSecurity[13]. Este paquete contiene algunos métodos que pueden especificarse dentro del mismo HTML con los que se puede hacer que algunos elementos se muestren o no dependiendo de si el usuario está identificado o no.

Por ejemplo, en el caso de la cabecera, como se ha dicho en el punto 4.2., a la derecha se mostrarán cosas diferentes dependiendo de si el usuario está autenticado o no. Esto se consigue gracias a los métodos *isAuthenticated()* y *isAnonymous()* que se encuentran dentro de este paquete.

[13] https://www.thymeleaf.org/doc/articles/springsecurity.html

5.3.2. Estilo

Uno de los objetivos que nos planteamos al inicio era que la aplicación fuera usable y es por ello que se han aplicado algunos estilos para que sea más agradable.

Entre los diferentes estilos desarrollados podemos destacar algunos como el cambio de la fuente de la letra. En vez de dejar que se muestre la fuente por defecto, he usado una fuente de las que ofrece Google a través de su navegador de fuentes[14]. Para esta aplicación he decidido usar diferentes tamaños de fuente y estilo de la familia Quicksand.

Otro cambio de los estilos a destacar es el cambio de la forma ratón cuando pasamos por encima de un botón o como en algunos de los casos también cambia el color del botón para indicar que estás encima de él.

También cabe destacar el diseño de la página inicial, donde las últimas noticias creadas aparecen en un formato grande para poder ver su imagen y ver bien el titular y el subtítulo.

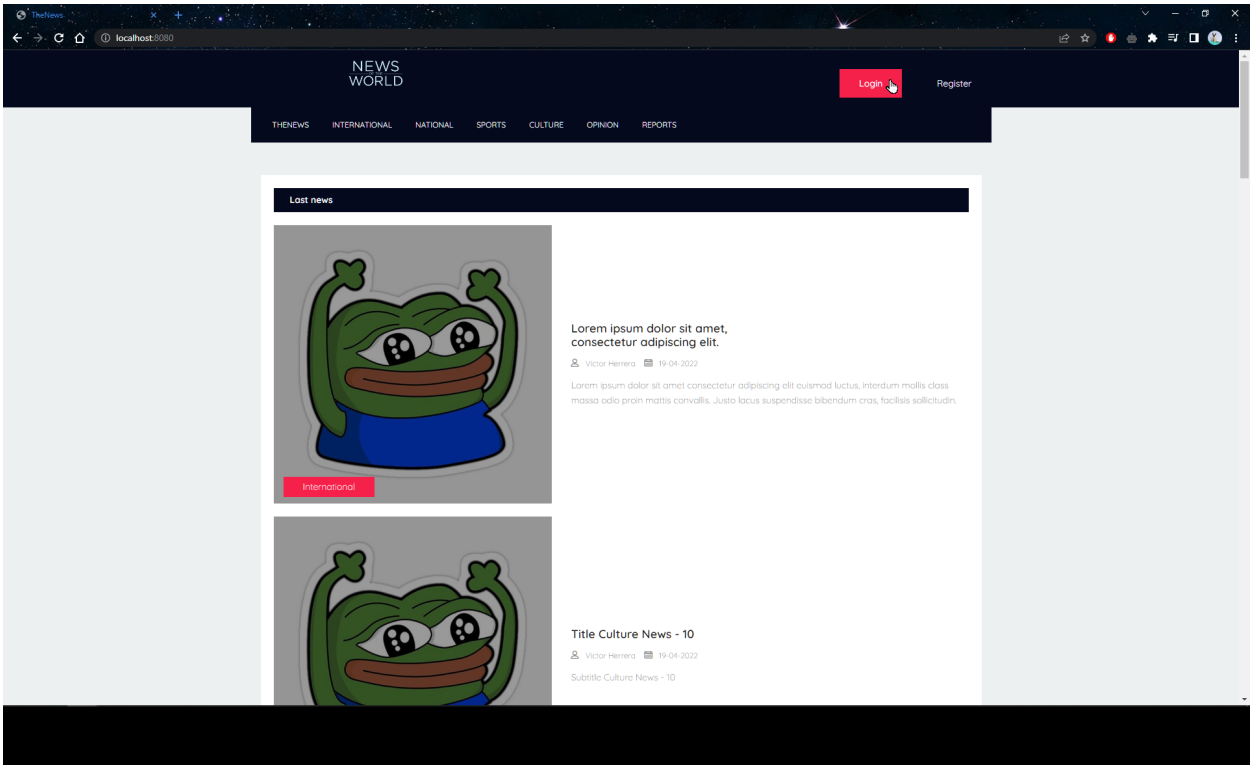

Figura 13. Botón cambiando de color y ratón cambiando de forma.

[14] https://fonts.google.com/

5.3.3. Pantalla de error

Una de las cosas que nos hemos encontrado a la hora de implementar el Frontend es que cuando se da con algún error del código, que hace que el navegador no sea capaz de interpretar lo que se quiere mostrar, muestra una página en blanco que resulta bastante molesta y que impide navegar a través de la aplicación, si no es con la ayuda del navegador dándole al botón de retroceder a la página anterior.

Es por eso que en el caso que esta situación se dé, hemos creado una página que controle el error y que muestre un mensaje de error genérico, así no rompe en excesividad la experiencia que pueda estar teniendo el usuario más allá de encontrarse con un error no deseado.

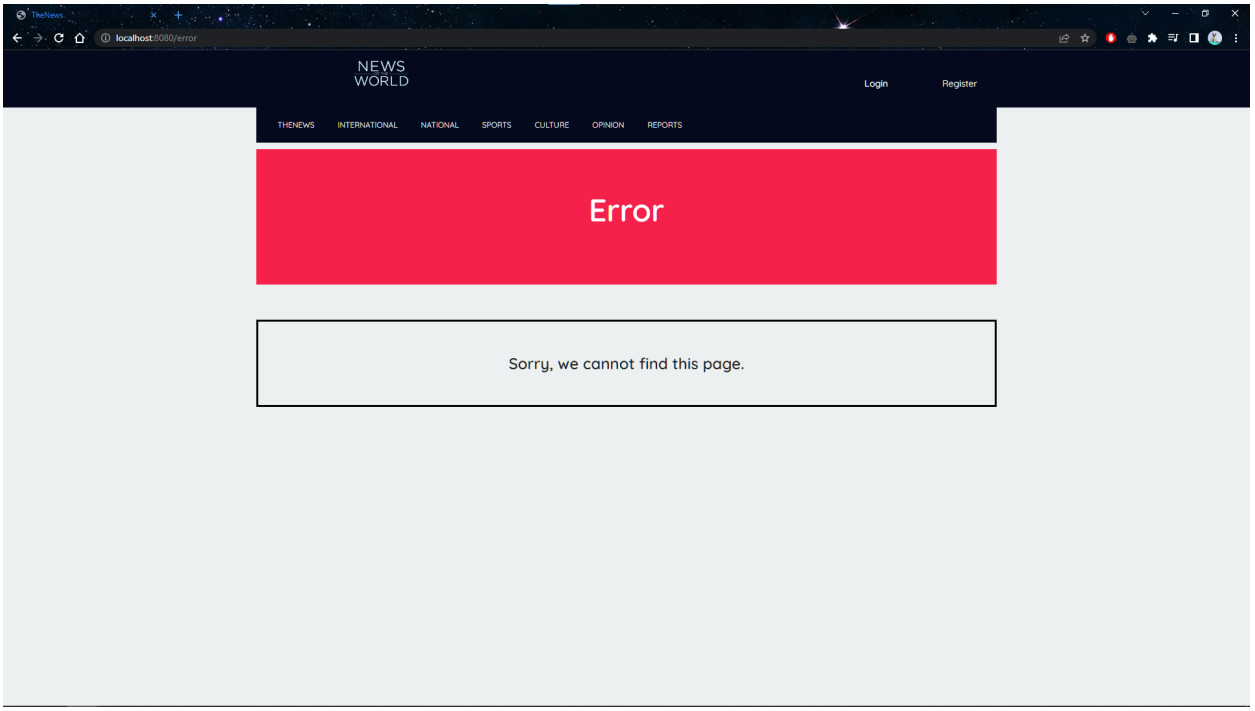

Figura 14. Página de error al fallar la aplicación.

5.4. Despliegue

En este punto, la aplicación ya está implementada y funcionando en mi máquina. Para dejar la aplicación lista para poder ser desplegada en algún servidor web si así se

deseara, hemos usado imágenes y contenedores de Docker para poder acceder a la aplicación, independientemente de cual sea el sistema operativo desde el cual se está lanzando la aplicación.

Así pues, hemos introducido dentro del POM de la aplicación el plugin *Docker Maven plugin* con el cual hemos creado la configuración para que se genere la imagen Docker de la aplicación. En la configuración, se especifica cual es la imagen base sobre la cual construir la aplicación. En nuestro caso es *openjdk:8-alpine*, y se indica que el ejecutable a incluir en la imagen es *thenews-0.0.1-SNAPSHOT.jar*. Este ejecutable se consigue al lanzar la instrucción *mvn clean package -DskipTests=true* en la terminal estando dentro del directorio donde se encuentra el proyecto en local.

Además, para hacer que las imágenes se usen en un contenedor, se ha creado en el archivo *docker-compose.yml* una red la que se crearán los dos contenedores necesarios, uno para la base de datos que usa la imagen de postgres del DockerHub público[15] y otro para la aplicación en el cual se usa la imagen creada anteriormente y que hasta que el contenedor de la base de datos no está levantado, no empieza a levantar el de la aplicación, puesto que si no fallaría.

[15] https://registry.hub.docker.com/_/postgres

6. Puesta en marcha

A continuación, veremos qué se necesita tener instalado para poder descargar la aplicación.

Para poner en marcha una copia local hay que seguir estos sencillos pasos.

6.1. Prerrequisitos

Para poder correr la aplicación primero debemos tener instalados en nuestro equipo local varios softwares.

El primer software, y el más esencial, es tener instalado Java JDK, cualquier versión igual o superior a la 1.8. Se puede descargar la versión más reciente desde la siguiente dirección: https://adoptium.net.

El siguiente software es Docker Desktop. Para instalarlo en nuestra máquina es suficiente acceder a su web y, según cual sea el sistema operativo, descargar el fichero o seguir las instrucciones indicadas. A continuación, se especifica como descargarlo:

- ➔ MAC: Descargar desde https://www.docker.com/products/docker-desktop
- ➔ Windows: Descargar desde https://www.docker.com/products/docker-desktop
- ➔ Linux: Descargar desde https://docs.docker.com/desktop/linux/install

También es necesario tener instalado el plugin Docker Compose. Ya tendremos este plugin en nuestra máquina si hemos instalado previamente Docker Desktop. Para más referencias se puede consultar esta dirección: https://docs.docker.com/compose/install

Otro software necesario es tener instalado Apache Maven. Para descargarlo e instalarlo, podemos seguir las instrucciones que se indican en la siguiente dirección: https://maven.apache.org/install.html.

6.2. Instalación y ejecución

Para la instalación de la propia aplicación y su ejecución hay que seguir los siguientes pasos:

1. Clonamos el repositorio con la siguiente instrucción desde el terminal del sistema: git clone https://github.com/realBikto/thenews.git.

2. Desde el mismo terminal del sistema, entramos dentro de la carpeta del proyecto que se ha clonado en la máquina y ejecutamos el script llamado *run_application.sh*. Este script hace tres instrucciones que, en caso que se tenga algún error, se pueden lanzar separadamente. Estas instrucciones son:

    ➔ mvn clean package -DskipTests=true. Con esta instrucción eliminamos los paquetes previos que se hayan creado y lo volvemos a generar, así tenemos la última versión del código y de datos.

    ➔ mvn docker:build. Gracias al plugin que hemos comentado en el punto 5.4. realizamos esta instrucción que construye la imagen de Docker de la aplicación.

    ➔ docker-compose up. Finalmente, levantamos la red definida en el archivo *docker-compose.yml* para ejecutar la aplicación.

3. Una vez la aplicación ya se haya iniciado, podemos acceder desde un navegador web con la dirección http://localhost:8080.

7. Conclusiones

En el apartado 1.2 se definieron unos objetivos a realizar para el proyecto. Después de llevar a cabo la implementación de la aplicación, podemos concluir que las condiciones y objetivos se han cumplido en la creación de la aplicación. En la aplicación, cualquier usuario que quiera, puede enterarse de cuáles son las últimas noticias del mundo, evitando las fake news. Además, cualquier usuario puede registrarse sin ningún coste y así poder escribir comentarios y existe la posibilidad de un usuario de tipo administrador que ve pantallas y puede realizar funciones que el resto de usuarios no.

Además, la aplicación se ha realizado con el uso de tecnologías habitualmente usadas en empresas dedicadas al desarrollo de software como servicio, lo cual era muy importante para mi de cara a un crecimiento profesional.

Gracias a cómo está implementada la aplicación, si en futuras versiones de la aplicación se precisa de más tablas o más datos que sean básicos para la aplicación, lo único que habrá que hacer será crear los archivos XML de Liquibase con los cambios que se han hecho en las tablas o en los datos que hay en esta primera versión. También, tal y como está implementado el Backend de la aplicación, si fuera necesario añadir los modelos de las nuevas tablas, así como añadir los servicios, controladores, etc., no romperá lo hecho previamente.

Debido a que en las fechas en las que estaba planificada la puesta en marcha de la aplicación hubo un problema en la sincronización entre Heroku y Github[16], a pesar que hemos conseguido desarrollar la web no hemos conseguido que sea accesible desde cualquier ordenador del mundo, puesto que para eso sería necesario que la aplicación estuviera en un servidor web. Al no estarlo, la aplicación sólo es accesible desde los ordenadores que cumplan los requisitos necesarios para instalar los programas comentados en el apartado 6.

En conclusión, el desarrollo de la aplicación no ha podido llevarse a cabo en su totalidad, pero las funcionalidades de esta primera versión de la aplicación han respetado las dos condiciones que impusimos en los objetivos.

[16] https://status.heroku.com/incidents/2413

8. Futuro trabajo

A lo largo de esta memoria, se ha comentado que esta es la primera versión de la aplicación y, por lo tanto, aunque es funcional, aún quedan muchas cosas por hacer para que sea más accesible aún y para mejorar la experiencia del usuario.

En la aplicación hay varias mejoras a realizar, como por ejemplo la creación de botones para acceder a todas las noticias que se han creado en un mismo día. En esta versión podemos ver quién es el creador del artículo así como la fecha, pero solo el redactor es un botón accionable. Sería interesante que pudiéramos ver las distintas noticias que se han creado el mismo día, así como añadir un buscador para poder usarlo de hemeroteca.

También podemos mejorar varios factores relacionados con el usuario. Lo primero sería desarrollar una pantalla de gestión de usuarios, para que cada usuario pueda editar su información, cambiar su contraseña o actualizar su correo. Referente a esto último, habría que mejorar la seguridad de la aplicación y del sistema que lo gestiona haciendo que cada usuario que se crea no se active hasta que verifique su correo, evitando así la creación de múltiples cuentas falsas.

Otra de las cosas que nos hemos dado cuenta que hay que mejorar mientras se implementaba la aplicación es la necesidad de la creación de, al menos, un rol más llamado “EDITOR” que permita la creación de artículos, sin permitir que pueda cambiar el usuario que crea el artículos e indicando que es el mismo usuario que está registrado, y la edición de los artículos en los que el creador sea el propio editor.

Finalmente, como hemos comentado en las conclusiones, uno de los trabajos que han quedado pendientes en este proyecto y que habría que realizar es la puesta en marcha on-line de la aplicación así como las pruebas de usuario para averiguar si su experiencia es lo suficientemente agradable y usable.

9. Bibliografía

CSS Google Fonts. W3schools. https://www.w3schools.com/css/css_font_google.asp

CSS Tutorial. W3schools. https://www.w3schools.com:443/css/

Get Started with the Google Fonts API. Google Developers. https://developers.google.com/fonts/docs/getting_started

Huß, R. *fabric8io/docker-maven-plugin*. Docker-Maven-Plugin. https://dmp.fabric8.io/

Legacy container links. Docker Documentation. https://docs.docker.com/network/links/

Paraschiv, E. *Retrieve User Information in Spring Security*. Baeldung. https://www.baeldung.com/get-user-in-spring-security#get-the-user-in-thymeleaf

Spring Security | Baeldung. Baeldung. https://www.baeldung.com/category/spring/spring-security/

Spring Security with Thymeleaf. Baeldung. https://www.baeldung.com/spring-security-thymeleaf

Los periódicos más antiguos de España. My News. https://mynews.es/los-periodicos-mas-antiguos-de-espana

Historia de la prensa española. Wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Historia_de_la_prensa_espa%C3%B1ola

Nacimiento y Consolidación de un Sector en Auge. Razón y Palabra. http://www.razonypalabra.org.mx/anteriores/n47/gomezpaniagua.html#:%7E:text=1994%20marca%20el%20comienzo%20de,de%20las%20posibilidades%20de%20Internet

Era digital. Economipedia. https://economipedia.com/definiciones/era-digital.html

Los primeros periódicos digitales. Soso's blog. https://soso.blogs.uv.es/2009/11/24/los-primeros-periodicos-digitales/