

CFGS DESARROLLO DE APLICACIONES WEB ONLINE



GENERALITAT
VALENCIANA

Conselleria d'Educació,
Cultura i Esport



Florida

Universitària

Proyecto Final de Ciclo

Memigo: Web Edition 2023-2024



Apellidos y nombre del autor/a: Roberto Martínez Avendaño

Fecha de entrega: 17/05/2024

Contenido

¿Qué es Memigo?	3
Propuesta del proyecto	3
Desarrollo del proyecto	5
1.Estudio de mercado	5
2.Metodología usada	7
3.1.Tecnologías utilizadas	7
Base de datos	7
Back-End: API	8
Front End: Diseño	8
Front End: Interactividad	8
Front End: Securización	8
3.2.Wirfeframe, Mock Ups y Diseño de BBDD	9
Diseño de Base de Datos	9
Wireframe y Mock Up	10
3.3.Componentes de la aplicación: Back-End y Front-End	17
Back-End	17
Front-End	19
3.4.Problemas encontrados durante el desarrollo	22
3.5.Resultados obtenidos	23
Conclusiones	24
Líneas futuras de trabajo	24
Bibliografía	25

¿Qué es Memigo?

Memigo es una aplicación web diseñada para funcionar como una red social, donde los distintos usuarios pueden crear o compartir sus propios memes. A lo largo de la historia de Internet, los memes han sido una parte muy importante para esta, debido a ser uno de los contenidos más consumidos por todas las edades. Existen muchos tipos de memes, en todo tipo de formatos, al igual que hay muchas aplicaciones o herramientas dedicadas a la creación de estos, desde el más simple meme hecho en Paint, hasta algunos programas de edición que van más allá de solo poner texto sobre una imagen graciosa. También hay muchas aplicaciones o webs destinadas a compartir estas imágenes, ya sean páginas de Facebook, cuentas de Instagram, foros o webs como [CuantoCabron](#) o [CuantaRazon](#), entre otras.



Memigo busca insertarse en el mercado como una aplicación sencilla que te permita cumplir ese objetivo: **Crear, Compartir y Descargar** memes, ya sean los tuyos propios creados con la aplicación o los que más graciosos te parezcan hechos por otros usuarios.

Propuesta del proyecto

Como se mencionó previamente, los memes llevan existiendo por internet ya muchas décadas. Este formato de contenido siempre se ha usado para expresar sátira o ironía haciendo uso de contenido audiovisual, un tipo de multimedia no siempre bien recibida, pero que inevitablemente cualquier persona que navegue por la red se ha topado tarde o temprano.



Nicolas Cage en Vampire's Kiss (1968), origen del meme "No me digas."

Al igual que el ser humano, el meme también ha ido cambiando y evolucionando hasta lo que es hoy en día, pasando por diferentes periodos o etapas, y teniendo diferentes públicos objetivo. En un inicio, lo más común era ver estos memes en páginas como Foros, donde los usuarios compartían sus propios memes entre ellos. Más tarde, se crearon webs dedicadas exclusivamente a este contenido, como ya se ha mencionado antes la famosísima página [CuantoCabron](#) donde se podían ver memes en español ya allá en la época dorada de los memes en el 2008. Sin embargo, esta clase de contenido multimedia no se quedaría solo ahí, expandiéndose a otras plataformas y redes sociales que estaban comenzando a ganar popularidad, como **Twitter, Youtube, Facebook o Instagram**.

Cualquier persona que no haya vivido en una cueva las últimas dos décadas conoce a la perfección lo que es un “Meme”, sin embargo, a medida que el internet fue creciendo y expandiéndose, también lo hizo con ello las formas de humor de las personas. Es así como los memes logran inundar toda la internet, y aprovechando el boom de estos, nacen las aplicaciones para crear memes. Un meme puede expresarse de cualquier manera, ya sea con un dibujo, o poniendo un rotulo a una imagen que pueda considerarse graciosa, o una música a un video que pegue con el contexto. Más esto requería de programas externos que un usuario promedio que solo quiere poner un texto gracioso a una imagen aleatoria.

Desarrollo del proyecto

1. Estudio de mercado

En el mercado existen muchas aplicaciones de memes, y no solo webs, si no también apps móviles, y otra decena de programas para crear este contenido multimedia. Es difícil diferenciarse del resto en un mundo tan establecido como es el de los Memes. No obstante, Memigo busca que sus usuarios tengan una experiencia rápida y simple, ya sea

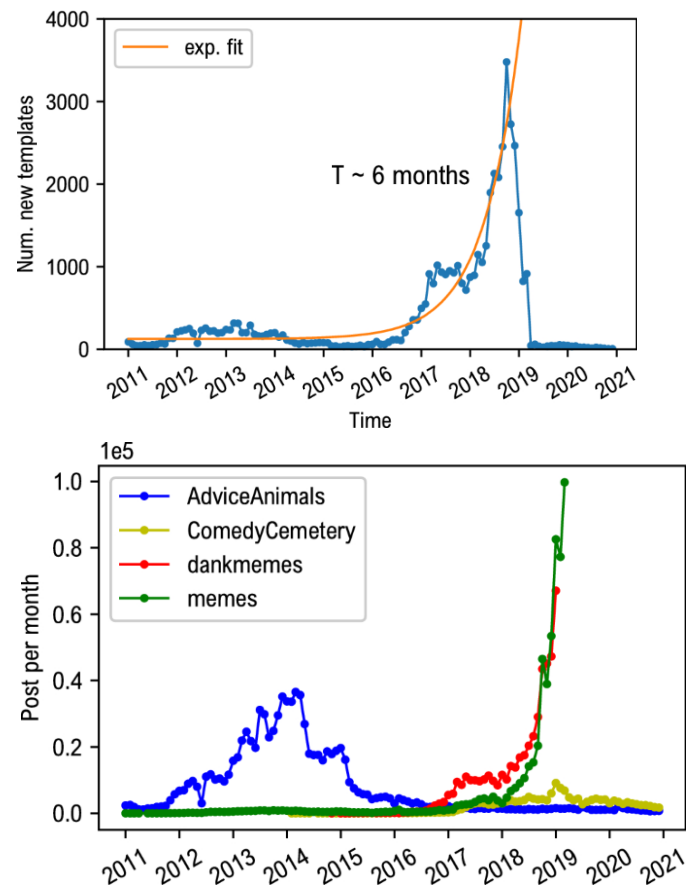


perdiendo decenas de horas navegando entre la página principal de Memes o bien pasando un buen rato creando y compartiendo con sus amigos sus propios Memes. La aplicación en sí, no busca ser una red social de Memes, aunque este es uno de los principales atractivos, sino más bien desarrollar un buen creador de Memes, simple pero atractivo, que no limite a sus usuarios, pero que tampoco sea muy engorroso. Crea un Meme, súbelo, descárgalo o simplemente descártalo y haz otro. Esta es la principal motivación del desarrollador principal y gran amante de los Memes, Roberto Martínez Avendaño (**Roberto Memes**).

Según un estudio realizado por la página web Nature en el año 2021, la producción de memes y plantillas se ha cuadruplicado en los últimos años debido a la exposición de las nuevas generaciones a las redes sociales. Esto se comprobó realizando un estudio de análisis de datos proporcionados a través de los memes publicados en Reddit desde el año 2011 hasta el 2020.

En este estudio además se comprueba lo importante que es el hecho de que dentro de la comunidad memera, un meme tenga cierto impacto mediático lo cual le puede permitir prevalecer por muchos más tiempo que otros. Tal como es el caso de la famosa frase proveniente del Call of Duty: Advanced Warfare, famoso juego donde se dio inicio al meme de “F en el Chat”, el cual hoy en día se ha convertido en un estilo moderno de los jóvenes para presentar respetos.

Fuente: [Artículo de la revista Nature publicado en 2021](#)



Conjunto de datos utilizado en este trabajo. Cada curva muestra el número de publicaciones por mes que se descargaron de Reddit. Cada publicación corresponde a una imagen. La cantidad total de memes descargados es de aproximadamente 2 millones.

Para la realización de la Base de Datos se ha utilizado MariaDB debido a que llevo varios años trabajando con este mismo lenguaje SQL de Base de datos. También he utilizado el editor de SQL HeidiSQL por ese mismo motivo.

Back-End: API



Para la realización de la API he decidido usar Java + Spring Framework debido a estar trabajando con estas tecnologías en las prácticas y contar con la experiencia para poder desempeñar esta tarea, además de ser muy cómoda y fácil de usar para crear APIs. Además de ello se ha añadido una securización de la API mediante el uso de JWT para lograr así una autenticación mediante Token.

Front End: Diseño



Para la realización del frontal de la aplicación he decidido utilizar Angular 17 con la implementación de Sass, Angular Material y Bootstrap 5 con la intención de hacerlo más atractivo visualmente y lograr un diseño completamente responsive.

Front End: Interactividad



A lo largo del curso hemos estado aprendiendo el uso de Typescript debido a ser el lenguaje que se usa junto con Angular para poder realizar la parte interactiva de sus webs. Es por eso mismo que al utilizar el anteriormente mencionado para la creación del proyecto, también he usado este lenguaje.

Front End: Securización

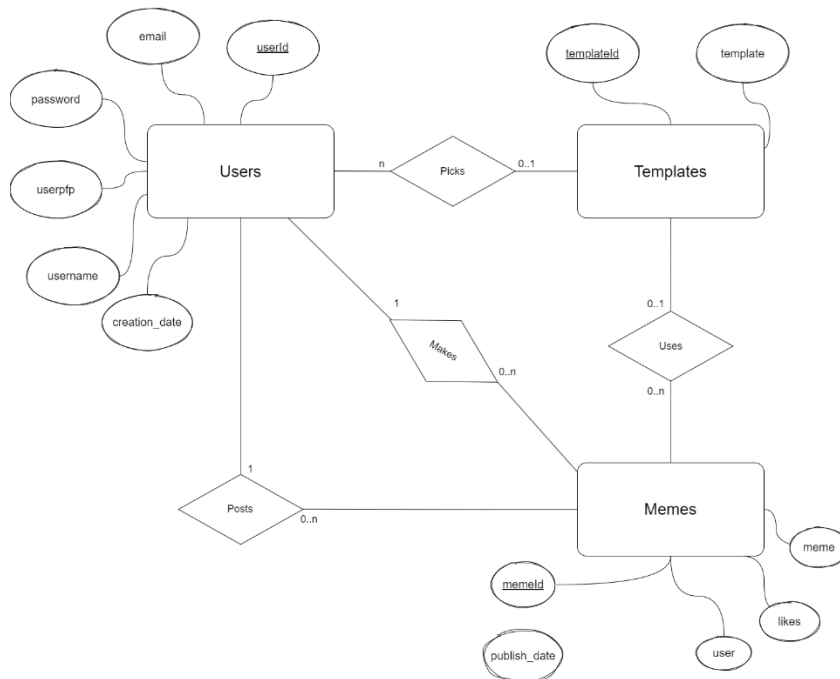


Para complementar la securización de la API y las llamadas a esta, se ha utilizado también JWT en los servicios de llamado a la API en Angular. De esta forma se ha logrado obtener un login que funciona mediante el uso de Tokens cifrados.

3.2. Wirframe, Mock Ups y Diseño de BBDD

Diseño de Base de Datos

MEMIGO - Design
Roberto Martinez



En el diseño original de la BBDD de Memigo reflejábamos la necesidad clara de la existencia de tres tablas: Memes, Plantillas y Usuarios, lo cual en esencia se mantuvo durante el desarrollo del proyecto. No obstante, tras contemplar la inclusión de JWT para la securización del API se generó la necesidad de una nueva tabla “Roles” y una tabla auxiliar para relacionar a Usuarios y roles. Además de ello, también se hicieron cambios en uno de los campos de la tabla memes, añadiendo el campo de “descripción” y cambiándolo por “fecha de publicación” mostrado en el esquema.

Wireframe y Mock Up

INICIAR SESION

Usuario o Correo
Contraseña

REGISTRARSE

He olvidado mi contraseña...

Iniciar Sesión

- Un usuario puede iniciar sesión con su @ o con su correo. Esto se comprueba con una expresión regular.
- Puede registrarse si no lo está. Si se ha olvidado su contraseña, se le manda un correo con un código para que pueda cambiar su contraseña. **Puede no estar incluido en el PMV*

Nueva contraseña

Confirmar contraseña

Introduce el código

Introduce tu correo

Originalmente el Wireframe hablaba de más funcionalidades de las que se pudo implementar al final de la aplicación. Muchas de estas se descartaron al llegar al Mock Up. Aquí podemos contemplar la posibilidad de cambiar contraseña mediante correo electrónico.

Iniciar Sesión

Registrarse

Originalmente también podías logearte o con correo o con usuario, de cara a la versión final esto se descartó, aunque queda para futuras implementaciones. El mock up al igual que el wireframe no contaban con un botón para “iniciar sesión”, esto obviamente fue un error del cual no me di cuenta. Pero en la aplicación final si hay.

REGISTRARSE

Usuario:

Nombre de usuario:

Email:

Contraseña:

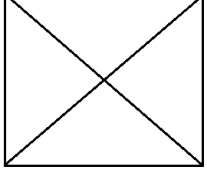
Confirmar contraseña:

Fecha de nacimiento:

Sexo: ☒ Si ☐ No

☐ He leído los terminos y condiciones de la aplicacion

Foto de perfil



Registrarse

REGISTRARSE

- Un formulario para registrarse
- Si el @ ya esta en uso en la Bbdd se avisa al usuario
- Si el correo esta ya registrado tambien se avisa al usuario
- La contraseña debe tener al menos una Mayuscula, minuscula y digito. Min. 8 caracteres.
- La opcion de Sexo esta sujeta a cambios en la version final.

Registrarse

Memigo 2


Usuario:

Nombre de usuario:

Email:


Contraseña:

Confirmar contraseña:

Fecha de nacimiento: 

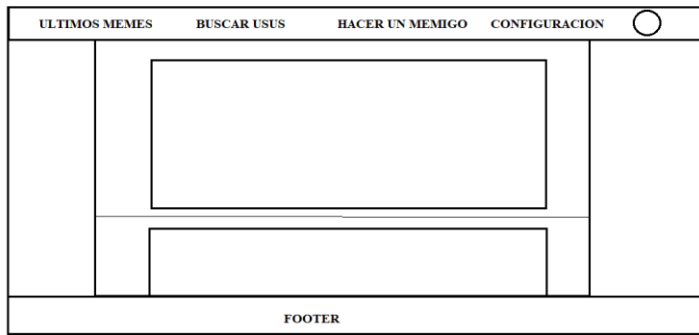
Género: ▼

☐ He leído y acepto los terminos y condiciones de la aplicacion



Registrarse

La pantalla de registro es probablemente la que menos cambios ha recibido de cara a la versión final. Se descartó la idea de introducir el género y fecha de nacimiento en el formulario ya que no son datos que formasen parte de los datos de usuario y no se tenia pensado ningún uso para ellos en la primera versión de la aplicación.



LA TL

Memigo Web es una aplicacion tipo Twitter. Las publicaciones se muestran de corrido en una timeline que carga todas las publicaciones a medida que se va scrolleando. Cada vez que se refresca la pagina, aparecen las publicaciones más recientes.

En buscar Usus se abre una pagina para buscar usuarios por su nombre o arroba y se muestran las coincidencias.

Hacer un memigo abre la pagina de hacer memes.

Configuracion abre la pagina de configuracion de web.

El icono de perfil te lleva a tu usuario.



Un meme:

- Imagen
- Una descripcion
- El corazon del me gusta
- numero de me gustas
- Boton para descargar
- Nombre del usuario



La página principal no recibió muchos detalles de cara al wireframe más allá del concepto de funcionar como la timeline de la aplicación de twitter cargando todo el contenido de memes. Además de esto aquí se puede contemplar la primera versión del Header con todas las funciones originales. De cara al Mock Up se creo un primer concepto de lo que sería un componente publicación con el nombre del usuario, la opción para likear un meme y descargarlo.

ULTIMOS MEMES	BUSCAR USUS	HACER UN MEMIGO	CONFIGURACION	
	Buscar un usu: <input type="text"/> <div>Coincidencias</div>			
FOOTER				

LA TL

Si no hay coincidencias a la hora de buscar un Usu, en el menu de coincidencias se desplegara un mensaje tipo "No hay informacion de eventos"

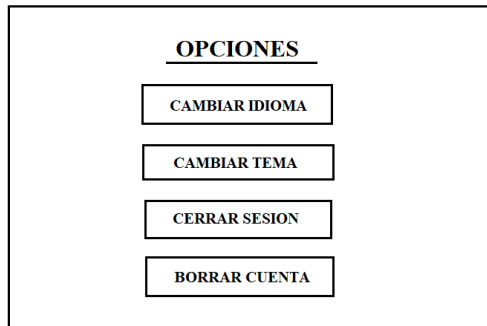
ULTIMOS MEMES	BUSCAR USUARIOS	HACER UN MEMIGO	⚙️	👤
<div>BUSCAR USUARIOS</div> <input type="text" value="@Paparopoulos18"/>				
<div>  <div> Papote Malote @Paparopoulos18 </div> </div>				
MEMIGO © ROBERTO MARTINEZ				

La función de buscar usuarios originalmente iba a contar con una pagina completa para todo este proceso. No obstante, esta idea acabó descartándose debido a la complejidad para imprimir los componentes usuario. En su lugar se acabó utilizando una barra de búsqueda de Angular Material.

Hacer Memes

- Elegir una plantilla de las disponibles en la BBDD
- Subir mi imagen. Aparece una modal para que recortes la imagen a tu gusto con el tamaño estándar de los memes
- Texto 1 y 2 para los textos de arriba y abajo
- Un botón para descargar y otro para publicar

Para la pantalla de hacer memes siempre estuvo contemplada la posibilidad de contar con un modal para poder recortar las imágenes y que cumplieren con el estándar de 500x500 de los memes. De cara a la versión final lo que más vario fueron las plantillas, ya que al final se usaron otras 5. Pero esta planeado que a futuro se cuenten con más plantillas, por eso en el mock up y el wireframe se puede ver unas flechas junto a las plantillas.

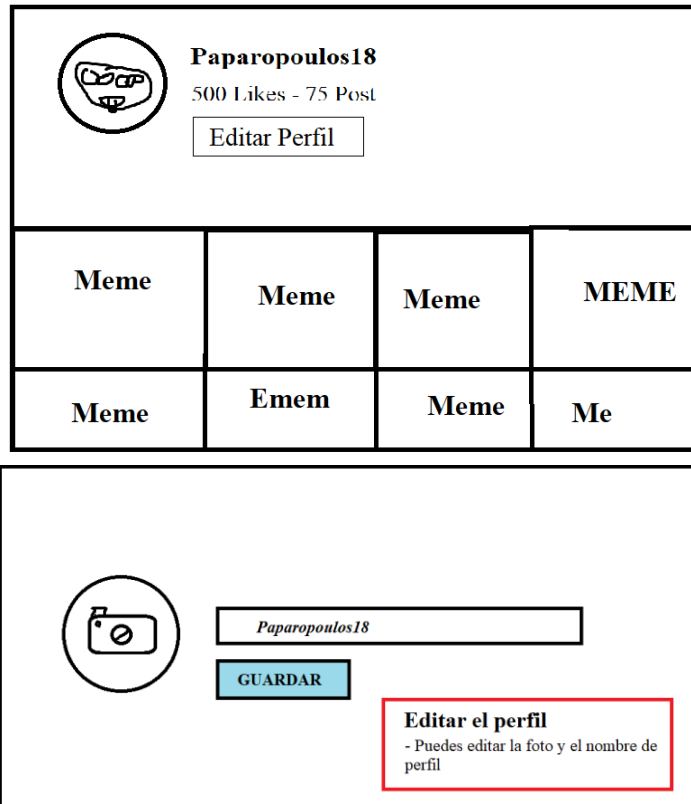


Opciones

- Cambia el idioma entre español e ingles
- Cambia el tema entre Memigo Clasico (Paleta habitual) y Memigo ReMeme (Paleta alternativa)
- Cerrar sesion es bastante autoexplicativo
- Borrar cuenta elimina la cuenta de la base de datos



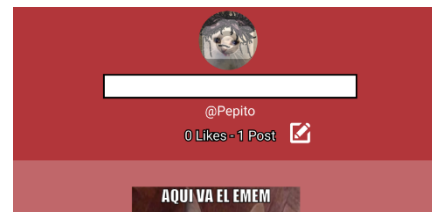
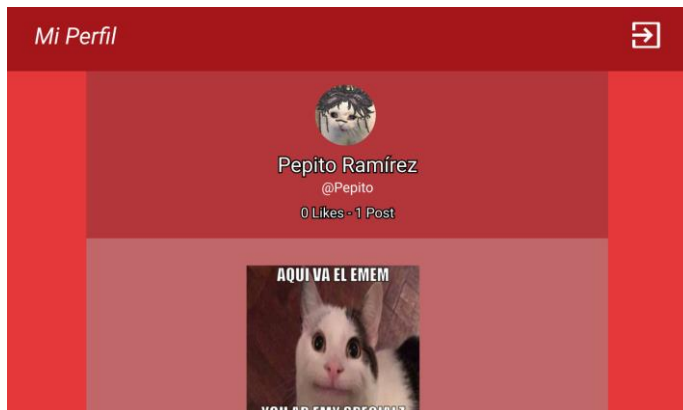
La pantalla de configuración iba a contar con una función de cambiar el tamaño de la letra para cumplir con accesibilidad, no obstante, la versión final es más fiel al Wireframe contando con solo 4 botones simples en línea con las diferentes opciones. El cambio de tema e idioma se tuvo contemplado desde el inicio. Otra diferencia a tener en cuenta con la versión final es que el encabezado de configuración no cambia a diferencia de como se muestra en el mock up, permaneciendo la barra de navegación en toda la página con excepción del formulario de registro e inicio de sesión



Un perfil

- Nombre de usuario y foto
- Recuento de likes y publicaciones
- Si es tu usuario "editar perfil" para cambiar la foto y el nombre de usuario
- Listado con todas las publicaciones (en debate si hacer modo galeria como en la imagen o modo scroll como la TL)

En el Wireframe se contemplo la idea de un perfil más parecido al de la aplicación móvil, mostrando en rejilla todos los memes y al clicar en uno te abriese un modal con el componente publicación con la información del meme seleccionado. La edición de perfil te llevaría a una pagina a parte para poder cambiar tu nombre y foto.



En la version del Mock Up ya se contempló algo más tipo Twitter, siendo que en la edición el nombre y la foto cambiaban por campos de formulario. Aunque en la version final se acabó dejando el editar perfil como una pantalla modal con los campos de foto y nombre.

3.3.Componentes de la aplicación: Back-End y Front-End

Back-End

/Models

- **Meme**
- **Role**
- **User**
- **Template**

Carpeta que cuenta con los archivos que tienen las entidades de la base de datos. Para realizar la conexión y la generación de código se usan las librerías de Jakarta y Lombo (que genera constructor, getters, setters, etc) Con el objetivo de ahorrar código y poder conectar la entidad con su tabla correspondiente en la clase de datos.

/DTOs

- **MemeDTO / MemeMapper**
- **UserDTO / UserMapper**
- **TemplateDTO / TemplateMapper**

Carpeta que cuenta con los archivos que tienen los objetos de transferencia de datos y su correspondiente mapeador el cual usando la librería de Mapstruck genera automáticamente el mapeado en la carpeta Target, pudiendo así tener la función de convertir de DTO a Entidad y Entidad a DTO sin tener que programar nada.

/Repositories

- **MemeRepository**
- **RoleRepository**
- **UserRepository**
- **TemplateRepository**

Carpeta que cuenta con los archivos que tienen los repositorios, estos utilizan la librería de JPA para poder utilizar todas las funciones predefinidas de un repositorio, además de ello hay algunas funciones personalizadas que se usaron para el tema de securización para buscar roles por nombre en RoleRepository.

/Services

- **MemeService**
 - *getAll, getByld, saveOrUpdate, delete*
- **TemplateService**
 - *getAll*
- **UserService**
 - *getAll, getByld, saveOrUpdate, delete*
- **UserDetailsServiceImpl**
 - *loadUserByUsername*

Carpeta que cuenta con los archivos que tienen los servicios, en estas clases están las implementaciones de los métodos de los repositorios para poder tratar los objetos que se mandan y se reciben mediante comunicación con la base de datos. Cada uno con sus propios métodos a nivel crud. Además, se ha implementado UserDetailsService para la securización con JWT.

/Controllers

- **MemeController + Interface**
 - *getAll, getByld, getMemesByUser, save, delete*
- **TemplateController + Interface**
 - *getAll*
- **UserController + Interface**
 - *getAll, getUser, getUserByUID, getUserByName, save, update, delete*

Carpeta que cuenta con los archivos que tienen los controladores y sus interfaces para poder implementar sus correspondientes métodos. Cada uno de los controladores son RestControllers que implementan los métodos de Spring y cuentan con sus correspondientes llamadas CRUD.

/Security

/Filter

- **JwtAuthenticationFilter**
- **JwtValidationFilter**
- **CustomUserDetails**
- **SimpleGrantedAuthorityJsonCreator**
- **SpringSecurityConfig**
- **TokenJwtConfig**

Carpeta con toda la lógica de la securización de JWT con los diferentes archivos, filtros, e implementaciones de las librerías usadas para generar los Tokens para la aplicación.

Front-End

Vistas

Usuarios: Modulo con todas las vistas referentes al usuario y su interacción con la Web. Aquí podemos encontrar los formularios de Login y Registro, así como la pagina de perfil y configuración.

- **Login:** Vista que cuenta con el formulario de inicio de sesión. Es bastante simple ya que cuenta con dos textboxes que usan un formulario con componentes de angular material, además de un hipervínculo a la pagina de registro y un botón para iniciar sesión. En caso de no introducir adecuadamente los datos de inicio de sesión saltara un modal.
- **Registro:** Vista que cuenta con el formulario de registro, tal como se comentó en el apartado de [Wireframe y Mock Up](#), se descartó la idea de pedir el sexo y la fecha de nacimiento. Si los datos introducidos no son validos saltara un modal avisando de esto, y hasta que no se acepten los términos y condiciones no se habilitara el botón de registro.
- **Perfil:** En esta Vista se encuentra el perfil de usuario en el cual se puede ver sus datos básicos, su numero de publicaciones y me gustas acumulados. Además, debajo se renderizarán todas sus publicaciones y en caso de no tener se mostrará un mensaje avisando de esto. Si el usuario es el mismo que ha iniciado sesión, se podrá editar el perfil.
- **Configuración:** Pantalla muy simple con un total de cuatro botones en línea recta los cuales ofrecen la posibilidad de cambiar tema e idioma u de cerrar sesión o borrar cuenta.

Memes: Modulo con todas las vistas referentes a los memes y su interacción con la Web. Aquí podemos encontrar la pantalla principal de Home y la aclamada pantalla para crear memes.

- **Home:** Vista donde se renderizan componentes Post con todos los memes de la base de datos, se contempló la posibilidad de añadir una paginación para no sobresaturarla de elementos, pero para la primera versión se decidió dejarlo para futuras implementaciones.
- **Meme Maker:** La mayor novedad y función más esperada de Memigo, una vista que da la posibilidad de crear memes introduciendo texto en las cajas de texto al lado derecho, así como elegir tus propias fotos o plantillas disponibles para crear estos. Al lado izquierdo se encuentra el meme y unos botones para subir el meme como publicación o descargarlo en tu dispositivo.

Componentes

Footer: Pie de pagina de la web, cuenta con enlaces a mi GitHub y LinkedIn además del nombre de la aplicación con el símbolo de copyright. El footer es sticky y responsive.

Header: Barra de navegación de la web, cuenta con el icono de la mascota (o *mascotas*) insignia de Memigo acompañado de los enlaces de navegación. También tiene una barra de búsqueda de usuarios implementada usando con angular material.

Dialogs: Se trata de una serie de pantallas modales con diferentes mensajes.

- **Delete Dialog:** Pantalla modal con un aviso respecto al borrado de la cuenta, con dos botones para saber si proceder o cancelar.
- **Login Dialog:** Pantalla modal que avisa que el login ha sido incorrecto.
- **Register Dialog:** Pantalla modal que avisa que el registro ha sido incorrecto.
- **Meme Dialog:** Pantalla modal que avisa que el meme se ha subido exitosamente.
- **Post Dialog:** Pantalla modal con un textarea para escribir la descripción de la publicación.

Post: Componente compuesto de una imagen con el meme y un par de encabezados con el nombre del usuario y su @ además del botón de los me gusta y el botón para descargar el meme de la publicación. Todo esto acompañado de una pequeña descripción.

Cropper-Modal: Componente que contiene el modal para recortar una foto previamente seleccionada. El modal cuenta con la pantalla para recortar la foto integrada gracias a la carpeta de ***npx-cropper-image*** y un par de botones para saber si se acepta o se cancela el recorte.

Servicios

- **Auth Service:** Servicio de autenticación con métodos para que funcione la securización de la API y así poder generar el Token.
- **Users Service:** Servicio con las llamadas securizadas a la API mediante el uso de JWT, también cuenta con métodos para obtener, subir, actualizar o borrar la información referente a los usuarios en la base de datos, además de métodos para obtener información del usuario logeado y así poder usarla en la propia aplicación.
- **Meme Service:** Servicio con llamadas a la API securizadas mediante el uso de JWT referentes a la tabla memes de la base de datos, con métodos para recoger y postear los memes/publicaciones directamente en la base de datos.
- **Template Services:** Servicio con llamadas a la API securizadas mediante el uso de JWT referentes a la tabla plantillas de la base de datos, con un método para recoger todas las plantillas de la base de datos.

Librerías usadas

- Ngx-Image-Cropper: Para poder crear la pantalla modal de recorte de imágenes.

Documentación de [ngx-image-cropper](#) en npm

- Ngx-Translate: Para poder traducir la aplicación entre español e inglés.

Documentación de [ngx-translate](#) en npm

- Angular Material: Para poder utilizar componentes ya creados de la librería material.

Documentación de [angular material](#) en su web oficial

3.4.Problemas encontrados durante el desarrollo

Durante el desarrollo de Memigo Web he tenido que enfrentar varios problemas, los cuales me ha llevado a tomar decisiones que han resultado en el descarte de algunas ideas originalmente pensadas para la aplicación. Los primeros problemas surgieron por la falta de tiempo, esto me llevó a descartar las siguientes ideas: Recuperación de contraseña mediante correo, Pagina de búsqueda de usuarios la cual fue sustituida por una searchbar en el header y por últimos la opción para cambiar entre tres tipos de tamaño de letra dentro de la aplicación. También mencionar una gran serie de problemas con los estilos, los cuales han sido mi mayor fuente de frustración por el diseño, ya que es uno de mis puntos flojos.

Otra serie de problemas surgió cuando quise implementar la securización mediante JWT en la web, al no tener un método que refresque el Token tras que este expire, al pasar cierto tiempo este caducará y la sesión colapsará, por lo que habrá que cerrar el servidor y volverlo a levantar para arreglar esto. Otro problema que enfrente con JWT fue que, al actualizar un usuario, la contraseña se volvía a cifrar por lo que la invalidaba. Esto fue solucionado corrigiéndolo en el método de SaveOrUpdate de la API en el servicio de Usuarios.

Un problema que me dejó muy exhausto fue el de poder guardar la información de las imágenes en base de datos. Ya de por si realizar el modal de crop fue bastante agotador debido a que gaste mucho tiempo en lograr que las imágenes se viesan adecuadamente. Para solucionar este error acabé aprovechando un método que tenia de la funcionalidad de hacer memes que desarrollé en los primeros días de proyecto, la cual capturaba la imagen y la guardaba como string en base 64. Como ultimo error más notable esta el de las funciones asíncronas, las cuales me generaron algún que otro dolor de cabeza, pero al final se pudieron solucionar ubicando las funciones entre try catches o llamándolas antes.

3.5.Resultados obtenidos

A pesar de las dificultades enfrentadas durante el desarrollo, he logrado obtener una primera versión de la aplicación con todo lo que me propuse que tuviese, no dejando ninguna pantalla descuidada ni tampoco pendiente de implementar. Lo que más prioricé por así decirlo, fue que la pantalla de creación de memes saliese a delante, ya que no pude implementarla en la aplicación de móvil el año pasado y lo sentí como un reto personal. La aplicación cuenta con un formulario de registro y un login funcional, que además cuenta con securización JWT que es algo que he aprendido durante mi periodo de FCT.

Además de ello, he podido solucionar más ágilmente los problemas que me he enfrentado, ya que, al disponer de componentes ya hechos en angular material, no he tenido que calentarme mucho la cabeza con la lógica de estos. Por último, la traducción también ha sido implementada lo último, de forma exitosa pese a encontrar algunos problemas, pero la experiencia previa del proyecto integrado hizo mi tarea mucho más sencilla.

Conclusiones

Después de haber estado trabajando de sol a sol durante varias semanas para poder sacar a delante este proyecto, he de admitir que ha sido una experiencia enriquecedora que me ha llevado a poner a prueba todo el conocimiento adquirido en estos tres cursos de DAM y DAW. He podido reencontrarme con JAVA el cual pensaba que era un lenguaje que no volvería a usar, he aprendido por mi cuenta como hacer una API Rest en condiciones y además he podido cumplir una promesa con Paco y conmigo mismo, desarrollando una aplicación para ver y hacer memes. También esta experiencia me ha ayudado a ver que estoy hecho para ser un desarrollador de Back-End en vez de Front-End ya que el diseño grafico definitivamente no es mi pasión.

El hecho de juntar este trabajo con las practicas, aunque en inicio ha resultado ser un poco molesto dado a que se solapaban los horarios y al acabar agotado de ir a la oficina, volvía sin ganas de programar. Toda la experiencia acumulada en mi itinerario de becario ha ayudado a que el proyecto se haga mucho más rápido ya que no solo se ha juntado el conocimiento con la experiencia, si no que me ha permitido salir delante de muchos problemas con una mentalidad más resolutiva al encontrarme con problemas que ya enfrente anteriormente. También poder reciclar código de proyectos pasados como el itinerario previamente mencionado u el proyecto integrado ha sido de gran ayuda.

En conclusión, desearía que este proyecto no se solapase con las practicas, aunque durase menos tiempo, pero como soñar es gratis, me quedo con que lo aprendido me ha servido para cumplir objetivos personales y desarrollar mi potencial como programador.

Líneas futuras de trabajo

Como futuras implementaciones a Memigo Web me gustaría rehacer el diseño de la web por completo, ya que es bastante fea en mi opinión. Pero como yo no soy diseñador gráfico, eso se lo tendré que dejar a otra persona. También esta el tema de todos los bugs que se me han quedado colgados, esos también habría que arreglarlos de cara al futuro. Añadir el refresco del Token para que la aplicación no colapse después de un rato y reforzar la securización de la Web, con mejores validaciones de errores, avisos más claros, etc. Queda mucho para que Memigo alcance su mejor versión.

Bibliografía

1. Componentes de Angular Material: <https://material.angular.io/components/categories>
2. Documentación de iconos de Bootstrap 5: <https://icons.getbootstrap.com>
3. Open AI Chat GPT para solución de errores: <https://chatgpt.com>
4. Trello para organización del proyecto: <https://trello.com>
5. Draw.io para el esquema de Entidad-Relación de la DDBB: <https://app.diagrams.net>
6. Figma para el Mock Up: <https://www.figma.com>
7. Florida Oberta, los apuntes de Paco para Angular.
8. Npm para descargar librerías de Angular: <https://www.npmjs.com>
9. Base 64 Guru para descifrar las imágenes en Base 64: <https://base64.guru/converter/decode/image>
10. Base 64-Encoder para codificar las imágenes a Base 64: <https://www.base64-image.de>
11. GitHub para mirar mi propio código y reciclarlo: <https://github.com>
12. StackOverflow como fuente de información ante algunas dudas y errores: <https://stackoverflow.com>
13. Documentación oficial de Bootstrap: <https://getbootstrap.com/docs/5.3/>