MEMIGO

proyecto de final de curso

Roberto martinez avendaño

2023

Tabla de contenido

[¿Qué es Memigo? 2](#_Toc135219747)

[Justificación del proyecto 3](#_Toc135219748)

[Propuesta 3](#_Toc135219749)

[Público y objetivo 4](#_Toc135219750)

[Metodología seguida 5](#_Toc135219751)

[Componentes de la Aplicación 5](#_Toc135219752)

[Mock Up original 5](#_Toc135219753)

[Diseño inicial de la BBDD 6](#_Toc135219754)

[Back-End: API Rest en .NET 6.0 7](#_Toc135219755)

[Tecnologías usadas 8](#_Toc135219756)

[Análisis de realización del proyecto 9](#_Toc135219757)

[Resultados obtenidos 16](#_Toc135219758)

[Funcionalidades de la aplicación desarrolladas: 16](#_Toc135219759)

[Funcionalidades de la aplicación pendientes: 16](#_Toc135219760)

[Conclusiones 17](#_Toc135219761)

[Bibliografía 18](#_Toc135219762)

# **¿Qué es Memigo?**

Memigo es una aplicación 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.

# **Justificación del proyecto**

## **Propuesta**

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. 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.

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

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](https://www.cuantocabron.com/) 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.

## Público y objetivo

En el mercado existen muchas aplicaciones memes, y no solo apps, si no también webs, 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).

Mujer con ropa deportiva

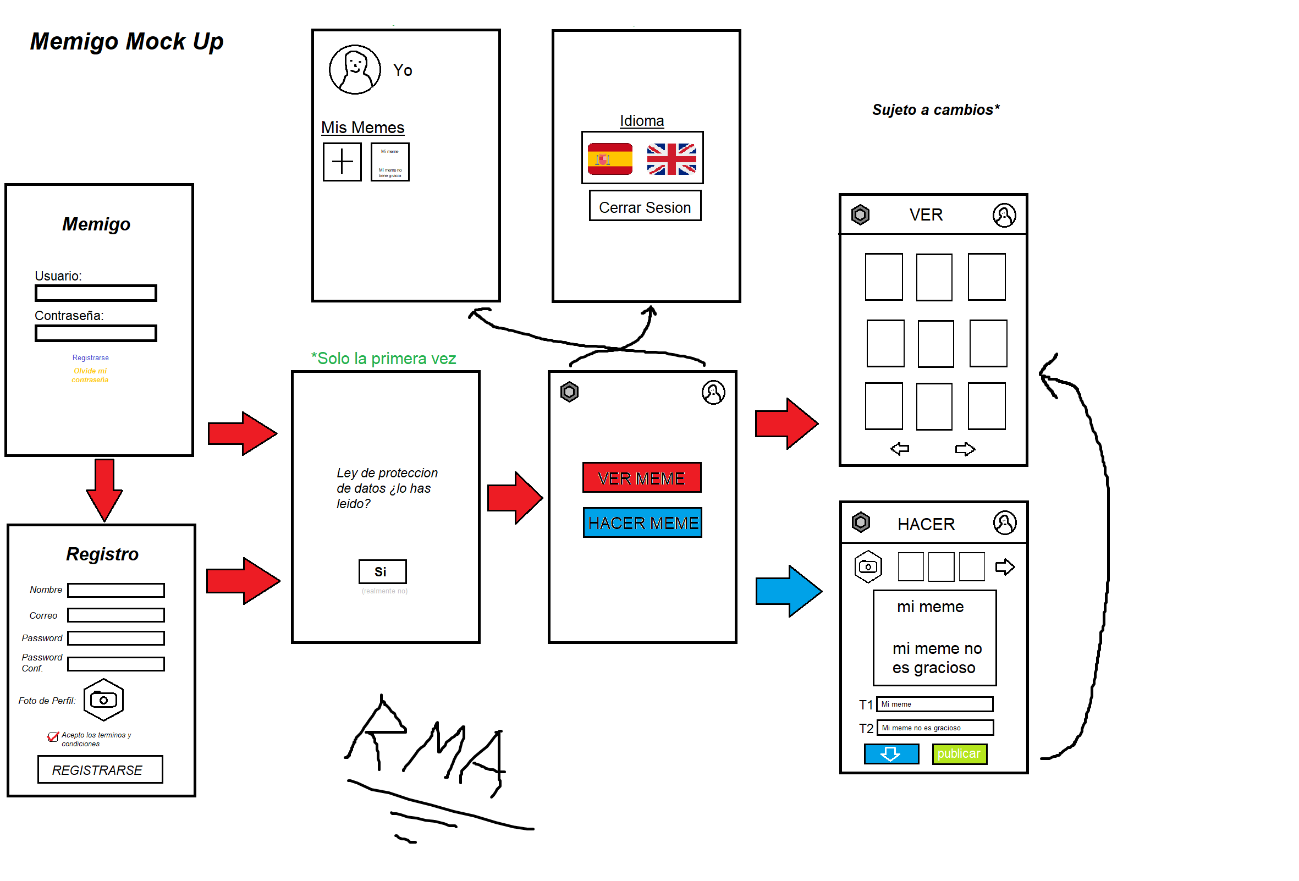
Descripción generada automáticamente con confianza media

# **Ventajas y desventajas de la metodología ScrumMetodología seguida**

El desarrollo del proyecto se a efectuado utilizando una metodología SCRUM, debido a que se a seccionando en varias entregas a lo largo de periodos de SPRINT durante el proyecto.

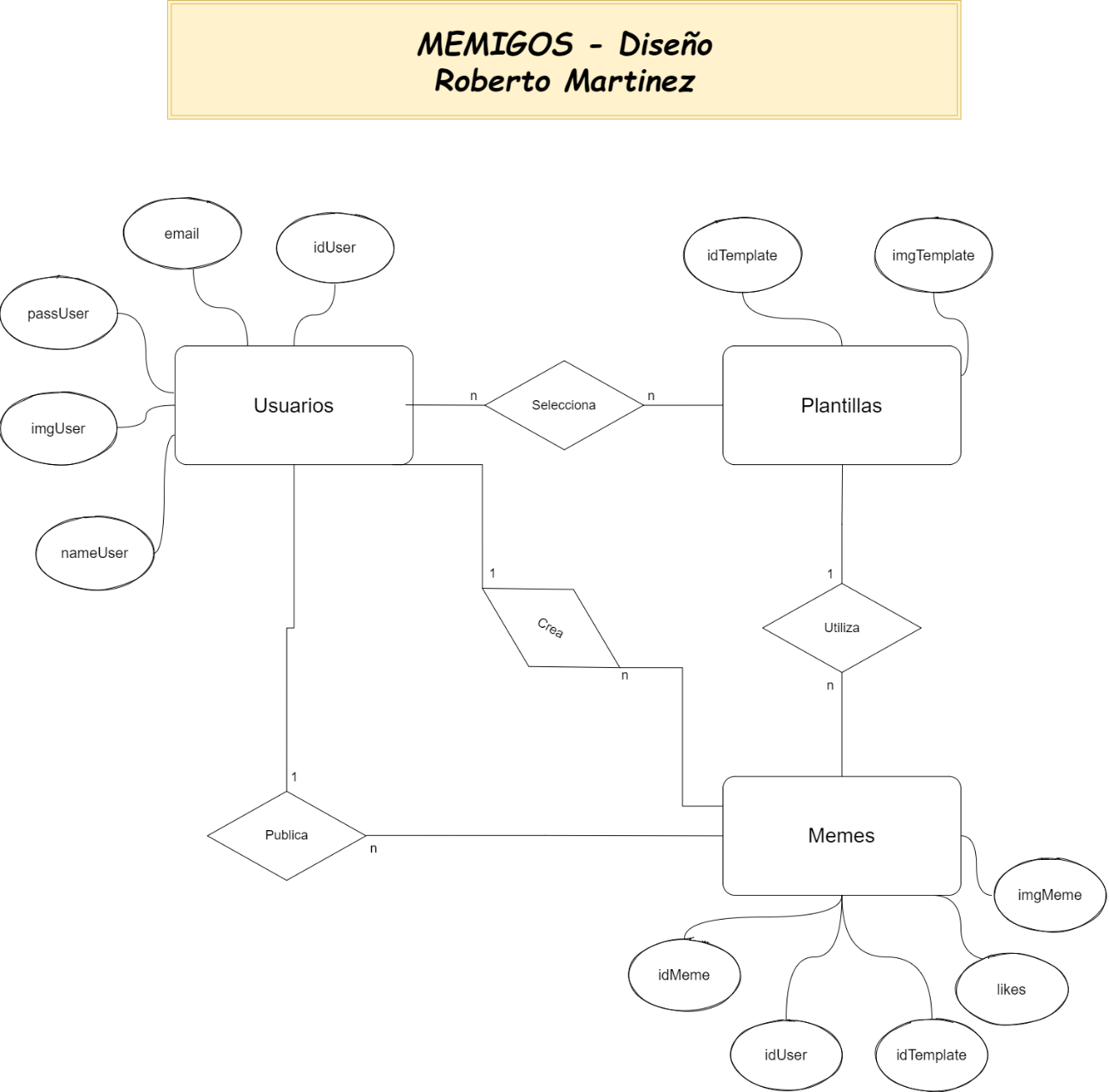
# **Componentes de la Aplicación**

## Mock Up original



La idea original de Memigo era crear una aplicación simple con la que poder crear tus propios memes, poder subirlos y también descargar los tuyos y los de otras personas. El diseño de la interfaz original iba a contar únicamente con dos botones en el menú ***(ver y crear meme)***, aunque esto fue más tarde actualizado a la hora de hacer el diseño final. Entre algunos de los cambios de diseño con respecto al Mock Up original, esta que, en el menú principal, la tuerca que serviría como acceso al menú de *“Opciones”* acabó cambiándose por un botón más en el menú principal. También se añadió la opción de *“Borrar Cuenta”* junto con el botón de *“Cerrar Sesión”*.

## Diseño inicial de la BBDD



La idea original de la base de datos de Memigo constaba de 3 tablas: Memes, Usuarios y Plantillas. Debido a que la implementación de la funcionalidad para ***“Crear Memes”*** se descartó, el diseño final de la BBDD no cuenta con la tabla de plantillas, donde se guardaba la información de las plantillas que los usuarios podían usar para crear sus propios memes. Conservando únicamente los Usuarios con todos sus datos de perfil y los Memes con todos sus datos de publicación. Para las imágenes se opto por trabajar con Base64 **(MEDIUMTEXT)** en lugar de fichero de imagen **(BLOB)** debido a haber trabajado más con ello a lo largo del curso.

## Back-End: API Rest en .NET 6.0



Durante el primer curso estuvimos trabajando en su mayoría usando C# con .NET, personalmente esta tecnología me intereso mucho más de lo que lo han hecho otros lenguajes, por lo que tras informarme de lo óptimo y sencillo que era levantar una API Rest con .NET, me puse manos a la obra.

Para el diseño del proyecto he hecho uso de una combinación de los patrones de diseño DTO, para la transferencia de objetos a la hora de hacer las peticiones a la API, y MVC, para tener una mejor organización del código.

Lo primero fue aprovechar las librerías de .NET para que una vez establecido el ***connection string*** con toda la información de la BBDD para que pudiese conectarse correctamente, generase la carpeta “**Model”** con el modelo de datos:

* *Meme.cs*
* *MemigoV1Context.cs*
* *User.cs*

Y dentro de esta carpeta, otra yo cree una subcarpeta **(DTO)** con sus correspondientes subcarpetas **(Meme y User),** donde se encuentran las clases para generar un objeto que pasarle a su correspondiente petición a la API.

**Model/DTO/User:**

* *AgregarUserDTO.cs*
* *ActualizarUserDTO.cs*
* *BorrarUserDTO.cs*

**Model/DTO/Meme:**

* *AgregarMemeDTO.cs*
* *ActualizarMemeDTO.cs*
* *BorrarMemeDTO.cs*

A dentro de la carpeta **“Controllers”** cree un archivo controlador para cada uno de las tablas de mi aplicación:

* *MemeController.cs*
* *UserController.cs*

En cada uno de los controladores, usando como único atributo un contexto de tipo ***readOnly***, y ayudándome de la documentación de .NET y la guía de un amigo mío que lleva varios años con esta tecnología le añadí a la clase el **[ApiController]** a cada controlador con su correspondiente ruta, antes de comenzar a generar las funciones de llamado a la API con sus correspondientes etiquetas **[Http]** en función de la necesidad *(GET, POST, PATCH, DELETE).*

Por último, en el archivo de lanzamiento añadí las políticas CORS para evitar problemas y un controlador para que ignore ciclos y así no crashee la API al cambiar la petición de *“usuarios”* a *“memes”.*

**IMPORTANTISIMO:** Para lograr que mi aplicación pudiese hacer peticiones a la API fue necesario cambiar la "applicationUrl" del archivo launchSettings.json por la IP de mi máquina. Al hacer la petición desde la App es muy importante pedir a HTTP y no a HTTPS ya que el protocolo de seguridad crea un error en la conexión.

## Tecnologías usadas

**Lenguajes de programación:**

* C#
* Javascript / React-Native
* MySQL

**Librerías y otras tecnologías:**

* .NET 6.0
* React-Native Paper
* Expo ImagePicker
* MariaDB

**IDEs:**

* Visual Studio 2022
* Visual Studio Code
* Expo-Snack
* HeidiSQL

# **Análisis de realización del proyecto**

**31/03 – Primera Reunión**

El día de hoy, tras volver a casa de las practicas, he tenido una primera reunión inicial con mi tutor para hablarle a cerca de la idea nueva que tuve para el Proyecto de Final de Ciclo.

Debido a un análisis de esfuerzo-recompensa con la previa idea de la aplicación de Quizzes, he tomado la decisión de cambiar a una aplicación más sencilla en términos de programación y de una temática que me apisona más, los memes.

Manel me ha pedido que le haga un MockUp/Wireframe completo explicando las utilidades de la aplicación con todas sus pantallas y se la envié. Por lo que está programado para que, en algún punto de la semana, aprovechando los días libres venideros, me ponga con el proyecto y el MockUp de este.

**06/04 – Creación del MockUp**

A lo largo de la semana me he marcado una serie de objetivos a cumplir del proyecto aprovechando mi tiempo libre, es por ello que el primero de ellos será realizar el **Mock Up** de la aplicación.

Nada más volver del trabajo, me he puesto en el ordenador para analizar que pantallas necesitara mi aplicación, aprovechando la explicación que le di en nuestra última reunión a Manel. Es probable que a futuro sean necesarias una o dos pantallas más, pero de momento se han planificado las siguientes*: Login, Registro, Menu, Ver Memes, Hacer Memes, Opciones y Perfil.*

La aplicación está pensada para poder cambiar el idioma de los textos entre español e inglés. Una vez acabado, se lo he enviado a Manel.

**07/04 – Planificación y creación de la Base de Datos**

Se ha decidido que para la creación de la Base de Datos se necesitaría una *BBDD* *relacional* creada en *SQL* que permita mantener la información unida entre las tablas. Una vez elegido el tipo de *BBDD* que se usaría para el proyecto, se ha contemplado el diseño de las tablas, haciéndose dos diagramas para concretar la información.

Tras pasar toda la mañana planificando sobre el diseño final de la base de datos, he creado finalmente en el *Heidi SQL* lo que sería la versión 1.0 de esta. La *BBDD* cuenta con estas tablas: *Usuarios, Memes y Plantillas*.

**Del 17/04 al 28/04 – Testing con Expo Snack, nuevas necesidades.**

A lo largo de estas dos semanas he estado aprovechando mis ratos libres en el trabajo para ponerme a experimentar con el **Expo-Snack**, he logrado comprender como funciona la biblioteca *“Image-Picker”* la cual está destinada a poder seleccionar una imagen del almacenamiento interno a elección del usuario.

Esta librería será útil de cara a programar el Editor de Memes interno de la aplicación. Tras algunas pruebas, he mirado otras librerías para poder añadir a estas imágenes seleccionadas desde el *“Image-Picker”,* texto para crear los memes. No obstante, parece ser que muchas de estas librerías no son compatibles con el editor de código de **Expo-Snack**.

Todo esto, sumado a la necesidad de exportar a **Android Studio** el proyecto para poder generar la APK al final del mismo, contando que mi equipo en casa no es capaz de correr el mencionado programa, me he visto en la necesidad de contactar a mis tutores para ver si la Florida puede suministrarme un equipo para hacer este trabajo.

Adicionalmente, ya que parecía que la pantalla de Edición de Memes estaría detenida un tiempo, me he puesto a desarrollar las primeras tres pantallas de la aplicación: ***Login, Registro y Menú***.

**01/05 – Pequeña actualización en la Base de Datos**

Aprovechando el día libre de hoy y tras pensarlo un poco, se han actualizado las tablas de la BBDD, cambiando el valor de las imágenes de Blob a String con la intención de tratar las imágenes en Base64.

**04/05 – Obtención del Portátil**

En la noche pasada recibí un correo de Manel diciendo que la Florida me había autorizado para poder pasar a recoger un portátil de la biblioteca. Después del trabajo he ido a por este y tras pasar por Sistemas para obtener permisos de administrador en el portátil para poder instalar todo lo necesario para realizar mi proyecto, he regresado a casa.

**05/05 – Avance en las pantallas**

A lo largo de la semana he estado aprovechando mis momentos muertos en el trabajo para continuar avanzando en **Expo-Snack** con las pantallas de la aplicación. Se ha creado la pantalla de ***Opciones***, con los botones de: *Cerrar Sesión, Borrar Cuenta y Cambio de Idioma*. También he creado la primera versión de la pantalla de ***Perfil de Usuario***, sin embargo, aún tengo que ver cómo solucionar el problema del *FlatList* con respecto a cómo se despliegan los memes en el perfil.

Además de ello se han creado comprobaciones para que en caso de que los datos estuviesen mal introducidos, salten advertencias al usuario. He encontrado algunas complicaciones con las *PantallasContext* para el uso de variables globales de estado por lo que le he solicitado a Paco una tutoría mañana.

**06/05 – Más problemas y menos soluciones**

Hoy durante la mañana tuve una pequeña tutoría con Paco para poder solventar mis problemas con el hook useContext de ReactNative, una vez solucionado ese problema, me dediqué a establecer las variables globales para las pantallas.

Durante la tarde tome el portátil para instalarle el Android Studio y así poder continuar experimentando con las interfaces en el portátil proporcionado por la Florida. Sin embargo, durante el proceso de instalación me di cuenta de que no tengo la virtualización activada en el portátil. Le he enviado a Manel un correo para comentárselo, estoy a le espera que me conteste, de igual manera tengo una reunión con él el lunes.

He creado en el Git del proyecto la rama SRC donde he puesto el proyecto de React en el que importare las pantallas hechas en Expo, por lo que hasta que se solucione el tema de la virtualización, esto permanecerá así.

**08/05 – Cambio de planes**

Después de salir del trabajo, me he dirigido a la Florida con el portátil para que me activasen la Virtualización, para ver si de esta manera podía utilizar el emulador de **Android Studio**. Sin embargo, tras ponerme junto a Manel al llegar a casa para ver si podíamos solventar mi problema, hemos comprobado que desgraciadamente no podía actualizar el driver.

Mis dos opciones actualmente son volver a ir a la Florida para pedir si se me puede activar el resto de cosas que requieren la virtualización, o descartar para un próximo sprint la funcionalidad de ***“Crear Meme”*** y continuar trabajando en el *Expo-Snack* para sacar el resto de pantallas que SI pueda hacer*.*

**09/05 – Progresos en Pantallas y API**

El día de hoy he avanzado bastante en el proyecto, habiendo añadido a la pantalla de registro la posibilidad de elegir una imagen de tu dispositivo para ponerla como foto de perfil. Actualmente, esta funcionalidad no parece estar funcionando de la mejor manera, sin embargo, el plan es convertir esa imagen de perfil en **Base64** y de ahí mandarla a la ***BBDD*** con un **POST**, por lo que aún se tiene que aplicar cambios a las pantallas. También se han hecho cambios estéticos en la aplicación *(Fondo y colores de los botones),* se ha completado lo pedido por Manel respecto al poder visualizar la contraseña, entre otros cambios menores. La pantalla de Usuario ya está operativa y se ha linkeado con el resto de pantallas.

Al regresar a casa he tomado el portátil y he instalado el Visual Studio 2022 con el objetivo de crear el proyecto **“API”**, cuya rama ya ha sido creada junto al proyecto en GitHub. Además de ello, previo a ponerme a trabajar con la API, tarea que dejo para mañana, he hecho a lo largo de la tarde diferentes cambios en el diseño de la base de datos:

* La tabla ***“Templates”*** se ha omitido de la BBDD debido a que la funcionalidad de ***“Crear Memes***” se ha aplazado para un próximo sprint
* Se ha añadido el **Borrado en Cascada** que no estaba presente en la primera versión, para que de borrar un usuario se borre con él sus memes.
* Se han creado diferentes Procedimientos Almacenados para que la API pueda utilizarlos como referentes a las acciones POST, GET, UPDATE y DELETE en la BBDD.

**10/05 – Icono en las pantallas, cambios hechos y API iniciada**

El día de hoy me he dedicado a arreglar algunos detalles menores en las pantallas, como el problema por el cual la imagen del perfil no encajaba en el círculo del avatar, o el error que daban algunas pantallas para linkearse entre ellas. También se ha completado la pantalla para poder editar el perfil, de modo que el usuario pueda cambiar su nombre de usuario y su foto de perfil.

Tras regresar el trabajo me he puesto con el proyecto de la **WebAPI** siguiendo un tutorial que encontré en Youtube y que se veía bastante prometedor. Sin embargo, tras avanzar lo suficiente me di cuenta que la API no arrancaba al no encontrar un archivo. El video utilizar **SQLServer** mientras que yo estoy usando **MariaDB** así que es probable que el problema resida ahí.

He pedido ayuda a un amigo especializado en **.NET**, mañana espero avanzar más en la API.

**11/05 – Cambios menores y API en estado avanzado**

El día de hoy he realizado arreglos de interconectividad entre pantallas para poder regresar desde *Options* hasta el *Menú principal*. Fuera de eso, no he hecho nada más orientado a Diseño de Interfaces. Mientras tanto, una vez llegue a casa, gracias a la ayuda de mi amigo, he revisado, aprendido y comprendido como realizar la **WebAPI** usando **EntityFramework**.

He corregido el fallo de la cadena de conexión, se han eliminado archivos innecesarios y usando el NuGet de Visual Studio 2022 he instalado librerías necesarias para facilitar el uso la programación en el proyecto.

Se ha conseguido sacar las peticiones *GET, POST, DELETE y UPDATE* para el apartado de la tabla **Usuarios**. Mañana seguiré por mi cuenta con la de la tabla **Memes**, y comenzare las pruebas de interconexión con la App. Preveo como muy tarde acabar este fin de semana el proyecto.

**12/05 – API quasi acabada, traducción completada, error de conexión.**

El día de hoy me he dedicado durante la tarde a traducir toda la aplicación para que, al cambiar a inglés, esta cambiase todos los rótulos y alertas de idioma. También he logrado sacar *el POST, UPDATE y DELETE* de **Memes** en la **API**, sin embargo, por alguna razón estoy recibiendo un error en el GET. Además de ello, aprovechando que el **GET** de usuarios está acabado, he intentado hacer fetch a la **API** desde Expo-snack, sin embargo, me ha devuelto un error de conexión. Mañana con más calma explorare una solución a este problema.

**13/05 – Adaptando el proyecto a local**

Durante la mañana me he dedicado a tratar de buscar una solución a los problemas encontrados ayer. Se me ha sugerido que para acceder en local a la API, el proyecto y la API deberían estar los dos conectados en la misma red. Por lo que, tras descansar apropiadamente durante la tarde, me he dedicado buena parte de la noche a migrar el proyecto de Snack a mi equipo, donde he logrado virtualizar la aplicación en mi dispositivo, quitándome de en medio la necesidad de usar Android Studio o tener que depender del emulador de Expo-Snack.

Una vez completada la migración he intentado acceder a la información de la API, no obstante, el error de conexión permanece. Me queda poco tiempo, y no estoy encontrando ninguna solución. Veré si puedo pedir ayuda a profesores o compañeros.

**14/05 – API Rest finalizada, FrontEnd prácticamente acabado.**

Durante la mañana he sido capaz de arreglar el error que generaba que no pudiese hacer una petición *GET* a la tabla de **Memes** en mi aplicación. Por lo que oficialmente la ***API Rest*** está acabada.

En vista de no poder solucionar el problema del Fetch a la base de datos, he estado re-factorizando el código del **FrontEnd**, y solucionando bugs o problemas que se me han ido presentando a lo largo de la mañana mientras llevaba a cabo diferentes testeos.

Como pendiente quedaría sustituir la información recibida en local por respuesta de la API Rest, sin embargo, al no poder lograr eso actualmente, estoy en un punto muerto. Tanto la **Base de Datos** como la **API** funcionan perfectamente. La aplicación es funcional. Pero al no poder conectar la **API** a la **Aplicación** no puedo hacer una simulación real.

Espero poder solucionar este problema pronto, no me queda mucho tiempo.

**15/05 – Interconectividad entre API y App, logradas mayoría de las funcionalidades.**

Ayer por la noche a última hora, gracias a la ayuda de mi compañero Victor, conseguí lograr que la aplicación hiciese fetch a la información en al **API**, por lo que hoy nada más volver del trabajo me he puesto a hacer las peticiones mediante métodos necesarios a la **API** para así poder interconectar las funciones con su correspondiente petición. La pantalla que más problemas me ha dado por el momento a sido la de ***Login***, se han presentado algunos problemas más a la hora de hacer los **Updates** de usuario con su correspondiente numero de publicaciones y likes, pero por el momento, se ha podido solucionar exitosamente.

No tengo idea de como hacer para que pueda mostrar los elementos de mi array cargado de memes en la pantalla de **Ver Memes** usando FlatList, tratare de pedirle ayuda a Manel para ver si me puede echar un cable mientras sigo buscando soluciones. Mañana se acaba.

**16/05 – Aplicación acabada, APK generada, muchos problemas, pocas soluciones**

El día de hoy nada más regresar del trabajo me he puesto para ver como solucionar el tema del renderizado del FlatList, en un poco rato he conseguido arreglarlo pudiendo mostrar todos los memes de todos los usuarios. Después de esto he estado viendo la manera de cargar la información del meme pulsado por el usuario en el componente **PostItem** de la pantalla Publicación.

Al final se pudieron solucionar todos los problemas de carga de imágenes haciendo una llamada asíncrona al método de carga. También se añadieron la funcionalidad de dar y quitar like a un meme en la pantalla de publicación. Mencionar adicionalmente que, a lo largo de la tarde, las pantallas de ***registro***, ***subir meme*** y ***login*** han estado dando diferentes errores, pero se han podido solventar.

Entre otros arreglos, se ha editado el código fuente del **Perfil de Usuario** debido a que los likes que tenia en total no se estaban reflejado, y se ha editado la FlatList del perfil de usuario para que muestre únicamente los memes publicados por el usuario.

Finalmente, tras descartar implementaciones de ultima hora, se ha generado el **APK** de la aplicación tras un buen rato buscando la forma de hacer esta.

# **Resultados obtenidos**

Con respecto a la idea original de la Aplicación, ya sea por falta de recursos o tiempo, se han tenido que hacer diferentes modificaciones para poder cumplir con el plazo de entrega. El desarrollo a sido un poco turbulento debido a tener que compaginar el proyecto con las practicas, que me han robado bastante tiempo, aunque he acabado aprovechando tiempo del trabajo para adelantar en mi proyecto. De no haber sido así, predigo que el Producto Mínimo Viable a entregar seria de una calidad mucho menor o quedarían pendientes muchas de las funcionalidades que al final se han podido obtener.

## Funcionalidades de la aplicación desarrolladas:

* Registrar a un usuario
* Entrar a la aplicación con tu usuario y contraseña
* Cambiar el idioma de la aplicación (español e inglés)
* Cerrar sesión
* Borrar tu cuenta
* Editar tu perfil de usuario (nombre y foto de perfil)
* Subir un Meme
* Ver un listado con todos los memes subidos
* Ver un meme en concreto subido a la aplicación
* Dar y quitar me gusta a una publicación\*
* Ver tu perfil de usuario con tu listado de memes subidos, numero de publicaciones total y me gusta total

## Funcionalidades de la aplicación pendientes:

* Activación de Alerts al introducir una contraseña o correo incorrectos
* Ver que usuario ha publicado un Meme al entrar en una publicación
* Borrar una publicación tuya
* Ver perfil de otro usuario de la aplicación
* Poder crear memes
* Permanencia sobre el like de un meme, para que cuando salgas y entres en una publicación donde diste Me Gusta, este se conserve y no se reinicie
* Poder descargar memes subidos de la App a tu dispositivo

# **Conclusiones**

El desarrollo de una aplicación móvil es todo un desafío que personalmente opino que requiere una gran inversión de tiempo. Durante el Proyecto Integrado, descubrimos esto ya que, en un periodo de dos semanas, entre un equipo de cinco personas, tuvimos que sacar adelante el proyecto, dedicándonos completamente en el tiempo de clase sumado a lo que pudiésemos hacer fuera del horario lectivo. Sin embargo, en este caso debíamos desarrollar enteramente un proyecto por nuestra cuenta, si bien el periodo de entrega era mucho más largo desde que se aceptaron las propuestas (finales de Marzo) hasta la fecha de entrega (mediados de Mayo), el tener que compaginarlo con las practicas y todos los proyectos en las que nuestra empresa nos pudiese involucrar, esto hace que el tiempo que se le pueda dedicar al proyecto sea mucho menor, y propuestas que puedan llegar a ser muy ambiciosas se queden a medio camino.

No obstante, esta es una experiencia muy valiosa ya que me ha permitido experimentar de primera mano lo que es el proceso completo de desarrollo individual de una aplicación. Ciertamente habría sido mucho más agradable contar con un poco más de tiempo para darle unos remaches finales a la App, sin embargo, he podido forzarme a buscar a resolver los problemas por mi mismo (con contadas excepciones) sin tener que depender de la ayuda de los demás.

También he podido aprender cosas nuevas como por ejemplo a levantar una Web API en .NET en lugar de Java, el uso de nuevas librerías y componentes para ReactNative, virtualizar mi aplicación en mi teléfono sin necesitar el Android Studio y su emulador, etc. Que la Florida me proveyese un Portátil para poder realizar mi proyecto también ha sido de crucial ayuda.

En definitiva, es una experiencia la cual me ha permitido ver las cosas desde un ámbito más profesional, a la vez que personalmente me enriquecía como programador.

# **Bibliografía**

* **Documentación oficial de .NET sobre Web API:** <https://learn.microsoft.com/es-es/aspnet/web-api/overview/getting-started-with-aspnet-web-api/creating-api-help-pages>
* **Documentación oficial de Expo:** <https://docs.expo.dev/>
* **Como generar una APK con Expo:** <https://dev.to/chinmaymhatre/how-to-generate-apk-using-react-native-expo-kae>
* **Expo-Image-Picker:** <https://www.npmjs.com/package/expo-image-picker>
* **Documentacion de React-Native Paper:** <https://callstack.github.io/react-native-paper/index.html>