UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Escuela de Ingeniería Informática

# Análisis, diseño e implementación de una aplicación para dispositivos móviles multiplataforma para la comunicación entre centros educativos y padres de alumnos II

ALUMNO
HÉCTOR FRANCISO GONZÁLEZ FEO
TUTOR
USUARIO DE WINDOWS

GRADO EN INGENIERÍA INFORMÁTICA (TECNOLOGÍAS DE LA INFORMACIÓN)
JULIO 2018 - LAS PALMAS DE GRAN CANARIA

Trabajo de Fin de Grado en Ingeniería Informática intensificación en Tecnologías de la Información de la Universidad de Las Palmas de Gran Canaria presentado por el alumno:

Héctor Francisco González Feo

**Título del proyecto**

Análisis, diseño e implementación de una aplicación para dispositivos móviles multiplataforma para la comunicación entre centros educativos y padres de alumnos II.

**Tutor**

Dr. Alexis Quesada Arencibia

# Agradecimientos

Aprovecho esta oportunidad para agradecer el apoyo ofrecido por mi tutor Dr. Alexis Quesada Arencibia, sacando tiempo para atender las dudas surgidas y prestarme su ayuda en todo lo posible para el desarrollo de este Trabajo Fin de Grado.

Como no, a mi compañero de viaje durante la carrera, Daniel Romero Calero, estando siempre ahí para levantarme en los momentos difíciles, y celebrar los logros conseguidos.

A mi pareja, a mis padres, a mis hermanos, y a toda mi familia, gracias a ellos soy quien soy y sólo puedo expresar mi más sincero agradecimiento por apoyarme durante la etapa académica que hoy culmina.

# Resumen

Este TFT proporciona una herramienta informática para facilitar la comunicación entre centros escolares y los tutores legales de los estudiantes. Actualmente, muchos centros siguen usando métodos convencionales para enviar todo tipo de mensajes informativos y comunicaciones (circulares, encuestas y autorizaciones). El empleo de estos métodos conlleva generalmente un gasto ingente de recursos (papel, tinta, …).

Para evitar los inconvenientes de los métodos tradicionales y agilizar estas comunicaciones, se ha desarrollado una aplicación móvil para la gestión de comunicaciones del centro que permitirá el intercambio de información de una manera más económica, fácil, segura y rápida. De esta forma, los tutores legales podrán recibir comunicaciones de la secretaría de los centros y de los profesores tutores, por ejemplo, para concretar una tutoría.

# Abstract

This TFT offers a tool to facilitate communication between schools and legal guardians of students. Currently, many centers continue to use conventional methods to send all kinds of informative messages and communications (circulars, polls and authorizations). The use of these methods usually involves a huge expenditure of resources (paper, ink, ...).

To avoid the inconveniences of traditional methods and speed up these communications, a mobile application has been developed for communications management of the center that will allow the exchange of information in a more economical, easy, safe and fast way. Thus, legal guardians can receive communications from the secretariat of the centers and from the tutor teachers, for example, to establish a tutorial.

# Índice

# Índice de ilustraciones

# Índice de tablas

# Introducción

Este Trabajo Fin de Grado (TFG) parte de un trabajo anterior titulado 'Análisis, diseño e implementación de una aplicación para dispositivos móviles multiplataforma para la comunicación entre centros educativos y padres de alumnos', realizado por D. Joel Delgado Perdomo y tutorizado por el Dr. Alexis Quesada Arencibia. Como ya se exponía en el anterior TFG, este proyecto pretende remplazar el sistema de comunicaciones que actualmente sigue siendo el más utilizado por la mayoría de centros educativos, el cual supone un problema. Por ello, hemos creado este proyecto que puede solucionar dichas dificultades, realizando un sistema mucho más sencillo, rápido, económico y que solventa la mayoría de problemas del sistema actual.

El trabajo consistirá en el desarrollo de una aplicación móvil y una plataforma web que servirá como medio de comunicación entre el centro y los padres. El desarrollo de la plataforma web será llevado a cabo en el TFG 'Análisis, diseño e implementación de un *backend* para la comunicación entre centros educativos y padres de alumnos II', realizado por D. Daniel Romero Calero y tutorizado por el Dr. Alexis Quesada Arencibia, y el desarrollo de la aplicación móvil, será desarrollado en este TFG.

La aplicación móvil a desarrollar contempla migrar todas las funcionalidades del estado anterior del proyecto, así como la implementación de un sistema de comunicación entre los padres y los profesores. Esta herramienta servirá para definir y gestionar los horarios de tutorías por parte del profesor y la petición de citas con los profesores por parte de los padres.

La aplicación web para el centro será la herramienta usada por los administradores de los centros para la realización de la gestión de los mismos. Tanto para el envío de mensajería de los padres, como para la gestión de cursos, alumnado y profesorado.

En esta fase del proyecto, se ha decido realizar cambios en la estructura anterior, además de añadir mejoras a las funcionalidades existentes y dar un paso más en el estado del proyecto. Se trata de un proyecto ambicioso y con gran potencial, ya que ayudaría a resolver muchos problemas en la gestión de las comunicaciones de los centros con los padres, pudiendo realizar una herramienta muy completa y de gran utilidad.

A partir de este punto, para referirnos a los términos, padre, madre, tutor o tutora legal usaremos padre o tutor legal, y para los términos profesor, profesora, utilizaremos profesor para estandarizar dichos términos.

# Estructura del documento

Esta memoria está separada por capítulos que procederemos a describir brevemente en las siguientes líneas:

1. Estado actual y objetivos iniciales.

   Establecemos el punto de partida del proyecto, detallando la situación actual del problema, estado del proyecto en la fase anterior, competidores de nuestra propuesta y los objetivos que nos hemos propuesto alcanzar con el proyecto.

2. Competencias específicas cubiertas.

   Enumeramos y justificamos cómo se han alcanzado las competencias cubiertas durante la realización del proyecto.

3. Aportaciones.

   Detallamos cuáles son las aportaciones que tiene nuestro proyecto en el apartado socioeconómico y el impacto personal que ha tenido en nosotros.

4. Normativa y legislación.

   Explicamos el análisis realizado de la normativa y legislación vigente y se incluye información sobre las licencias utilizadas de los diferentes productos software empleados durante el desarrollo del proyecto.

5. Metodología de trabajo y planificación del proyecto.

   Explicamos las características de la planificación elegida, así como el desarrollo de esta durante el desarrollo del trabajo final.

6. Tecnologías y herramientas utilizadas.

   Enumeramos y explicamos las tecnologías y herramientas utilizadas en el desarrollo del proyecto.

7. Análisis.

   Señalamos y detallamos el análisis realizado del proyecto, exponiendo las nuevas aportaciones y modificaciones de la fase anterior, así como la identificación de requisitos funcionales y la esquematización de estos mediante diagramas de casos de uso.

8. Diseño.

   En este capítulo definimos el diseño y la arquitectura bajo los cuales se ha construido el sistema. Además, estos se justifican debidamente y se detalla su implementación y funcionamiento.

9. Desarrollo.

   En este capítulo, explicaremos la estructura de los *frameworks* con los que se ha trabajado, así como las medidas tomadas durante el desarrollo que tengan especial relevancia para el desarrollo del proyecto.

10. Resultados, conclusiones y trabajo futuro.

    Exponemos los resultados del proyecto y objetivos cumplidos. Además, se exponen las conclusiones y posibles trabajos futuros a implementar.

11. Bibliografía

    Apartado con todas las fuentes de información y recursos consultados para la realización del proyecto.

12. Anexos

    Donde se encuentra el manual de usuario y el manual de instalación.

# 1. Estado actual y objetivos iniciales

## 1.1 Estado actual

### 1.1.1 Alternativas existentes

En primer lugar, antes de pasar al análisis del estado actual del proyecto, debemos analizar las necesidades que quiere cubrir este proyecto, así como el estado actual del que partimos. Además, comprobar los posibles competidores y soluciones existentes en el mercado, analizar si las alternativas disponibles satisfacen las necesidades del usuario y como ayudan a solventar el problema. Ya que, si conseguimos alternativas más eficientes a esas soluciones o encontramos algún factor a mejorar, tendremos más posibilidades de éxito de nuestro proyecto. Ayudándonos así a saber en qué tenemos que centrar nuestro desarrollo y seleccionar los mejores objetivos.

Teniendo en cuenta esta premisa, hemos decidido realizar un análisis del mercado actual para ver qué soluciones existen al problema de comunicación entre centros y padres haciendo uso de las nuevas tecnologías. A continuación, procederemos a enumerar dichas soluciones, así como a comentar que carencias encontramos en las mismas y que no existen en nuestro primer concepto de aplicación, o las ventajas de las que disponemos frente a su solución.

- **[1] FACTORYAPPS:**
  Es una empresa que se dedica al desarrollo de aplicaciones a media y ofrece un paquete de creación personalizada de aplicaciones para colegios.

  - **Ventajas:**
    1. Circulares y noticias exclusivas de las actividades extraescolares.
    2. Inscripciones a actividades extraescolares desde la aplicación.
    3. Integración de redes sociales.

  - **Desventajas:**
    1. Solo se puede tener la aplicación para un centro ya que desarrollan apps a medida para cada uno de ellos, impidiendo a los padres con hijos en diferentes centros mantener la información en una sola app.
    2. Solo gestiona información de actividades extraescolares y eventos, no facilita la comunicación interna del centro con los padres.
    3. No permite firmar autorizaciones.

- **[2][3][4] Tokapp School:**
  Es una aplicación cuya descarga es gratuita, pero que requiere de una suscripción por parte del centro para poder incorporarlo a la app y que puedan gestionar su contenido.

- **Ventajas:**
  1. Los datos del colegio se importan de manera automática.
  2. Gestiona los grupos del colegio y facilita la comunicación entre profesores y alumnos.
  3. Mensajería con caracteres ilimitados y con todo tipo de archivos.
  4. Los mensajes se confirman con la fecha y hora del momento de lectura.

- **Desventajas:**
  1. No permite la firma de autorizaciones.
  2. No dispone de un sistema propio de gestión de citas.

- **[5] miColegioApp:**
  Esta aplicación ofrece un canal de comunicación directo, inmediato y seguro entre centros escolares y familias. Esta aplicación además incorpora la opción de firmar autorizaciones.

  - **Ventajas:**
    1. Enviar circulares y documentos adjuntos.
    2. Aplicar filtros en el envío: alumno, grupo, curso, actividad extraescolar, comedor, etc.
    3. Recibir la información con notificación sonora.
    4. Confirmar citas, firmar autorizaciones, etc.

  - **Desventajas:**
    1. No incorpora la realización de encuestas.
    2. No dispone de auto-importación de datos.

- **[6] ClickEdu:**
  El software Clickedu es una plataforma escolar en la nube que incluye la gestión académica, administrativa y económica. Un entorno virtual de aprendizaje con conexión a libros digitales y contenidos gratuitos, gestión del profesorado, tutores y jefes de estudios, la calidad del centro y un entorno de comunicación con las familias, entre otros.

  - **Ventajas:**
    1. Esta app, de cara a los centros, ofrece una mayor facilidad a la gestión de recursos de los mismos, la contabilidad, etc.
    2. Permite la gestión de actividades extraescolares.
    3. Dispone de un aula virtual y contenidos digitales, notas, control de asistencia, herramientas de trabajo, acceso a contenidos digitales, evaluación competencial.
    4. Ver el calendario, el horario, las notas y la ficha del alumno.

  - **Desventajas:**
    1. No permite firmar autorizaciones.
    2. No incorpora la realización de encuestas.

3. No dispone de auto-importación de datos.

- **[7] Remind:**
  Esta aplicación se centra básicamente en la comunicación entre profesores, alumnos y padres. Es un canal de comunicación que permite mantener chats con los demás usuarios, ya sean alumnos con profesor, padre con profesores o padres con otros padres. Se crean grupos en forma de chats.

  - **Ventajas:**
    1. Creación de grupos de alumnos, padres y profesores en forma de chats.
    2. Envío de circulares.
    3. Comunica a padres y alumnos el mejor horario para contactar con el profesorado.
    4. Comprueba quién ha recibido y leído cada mensaje enviado.

  - **Desventajas:**
    1. No permite firmar autorizaciones.
    2. No incorpora la realización de encuestas.
    3. No tiene un sistema de auto-importación de datos.
    4. No dispone de un sistema de citas, solo incorpora el horario de oficina del centro y del profesorado.

- **[8] ClassDojo:**
  ClassDojo es una aplicación de comunicación para el aula. Conecta a profesores, padres y alumnos que la usan para compartir fotos, vídeos y mensajes durante la jornada escolar.

  - **Ventajas:**
    1. Permite al profesorado crear una red social con sus alumnos, valorar sus comentarios y enviar mensajes a sus alumnos en específico.
    2. Permite visualizar fotografías tomadas en clase o en actividades por parte de los padres. Actualizaciones por parte del profesor y comunicación privada con el profesorado.

  - **Desventajas:**
    1. No incluye envío de circulares.
    2. No dispone de encuestas.
    3. No permite firmar autorizaciones.
    4. No tiene un sistema de auto-importación de datos.

### 1.1.2 Estado actual del proyecto

Una vez analizados los posibles competidores, toca analizar el estado en el que se encuentra el proyecto y que necesidades cubre, que desventajas tenemos frente a las

otras soluciones. Además, definiremos nuevos objetivos para reforzar nuestro proyecto y diferenciarlo, ofreciendo una alternativa a los productos del mercado actual.

En la versión actual, encontramos la posibilidad de manejar diferentes centros desde una sola aplicación móvil. En ella se puede recibir tanto las circulares como autorizaciones de nuestros hijos, así como responder pequeñas encuestas que realicen los centros. Con ello, conseguimos un canal de comunicación directa entre el centro y los padres, eliminado el papel ya que el sistema está completamente automatizado.

Además, el sistema incorpora un sistema de auto-importación de datos, lo que permite una gestión mucho más sencilla para la puesta en marcha el sistema.

El hecho de contar con la firma de autorizaciones nos da una ventaja sobre la mayoría de los competidores, aun así, no es suficiente, ya que algunos de ellos también disponen de esta ventaja. Por ello, debemos reforzar el punto débil de nuestra propuesta, la comunicación padre/profesor. Para ello, implementaremos un sistema de gestión de citas de tutorías entre padres y profesores.

Dado que el principal objetivo que queremos alcanzar en este proyecto es mejorar la comunicación entre centro/profesores y padres, el objetivo de este Trabajo de Fin de Grado será mejorar los puntos débiles del proyecto frente a las demás soluciones del mercado y mejorar aspectos de seguridad sobre el proyecto anterior.

Respecto a la tecnología utilizada para el desarrollo de la aplicación también requiere implementar una mejora. Esto se debe a que actualmente se encuentra desarrollada con [9] *Ionic 1,* y la versión actual ha mejorado en diferentes factores como el rendimiento, la apariencia o la accesibilidad a elementos nativos del dispositivo.

### 1.1.3 Conclusiones

Como hemos podido observar, si recogemos todas las ventajas de cada aplicación para crear una, quedaría cubierta toda la funcionalidad que le queremos dar a nuestro sistema. Pero, si analizamos cada aplicación individualmente encontramos aspectos que se pueden llegar a mejorar ya que ninguna las cubre todas.

En las soluciones analizadas, encontramos diferentes tipos de aplicaciones:

- Las que están más centradas en la gestión del centro, organización de las clases y tareas, seguimiento de alumnos, deberes, menús de comedor, transporte escolar, actividades extraescolares, etc.
- Otro grupo trata de mejorar la comunicación, pero la mayoría da mayor importancia a los chats.

Nuestra solución lo que quiere conseguir es mejorar la comunicación con el centro y que sea muy sencilla de utilizar sin complicar su uso. Por lo que, la parte de gestión nos parece innecesaria en el proyecto.

Respecto a la mensajería, nuestra aplicación se centra más en la información general y de carácter oficial y permite suplir la mensajería individual que las alternativas ofrecen,

ya que toda la comunicación se realiza desde el centro y es este el que decide si desea una respuesta y de qué tipo específico la pretende.

En esta segunda iteración del proyecto vemos necesario añadir la gestión de citas con el profesorado, ya que, como se ha comentado, desde la aplicación móvil no se permite una comunicación por parte de los padres más allá de responder a las cuestiones o información proporcionada por el centro, pero sabemos la importancia y necesidad de mejorar la comunicación directa con los profesores de los alumnos y es una carencia de nuestro sistema frente a muchas de las otras soluciones.

Por último, ambos grupos cuentan con aplicaciones pensadas para ser utilizadas por padres acostumbrados a las interacciones más modernas pero que pueden ser no tan intuitivas. Nuestra solución brinda la facilidad de uso por su claridad y sencillez, intentando ser lo más intuitiva y clara posible.

## 1.2 Objetivos

El objetivo principal de nuestra solución es ofrecer una solución sencilla, amigable con el medioambiente y viable incluso para aquellos centros con infraestructuras mínimas y personal inexperto. Para ello se desarrollará una aplicación móvil que permita facilitar las comunicaciones entre el centro y los padres y profesores. Hasta ahora el sistema permite enviar circulares informativas, encuestas y autorizaciones de actividades extraescolares. En esta segunda fase, se desarrollará un sistema de gestión de tutorías entre padres y profesores.

En el caso específico de este Trabajo de Fin de Grado, los objetivos son claros:

- Actualizar el sistema existente a las nuevas tecnologías, mejorando así el rendimiento y la seguridad del sistema anterior.
- Ofrecer una interfaz de usuario sencilla, clara e intuitiva.
- Proporcionar siempre al usuario transparencia sobre qué acciones realiza en la aplicación.
- Garantizar la seguridad del usuario, la privacidad de sus datos y qué centros pueden verlos.
- Asegurarnos de que la información de los centros y la retroalimentación que estos requieren llegue a los padres y no se extravíe como podría suceder con métodos más tradicionales.
- Proporcionar un método de comunicación con los profesores por parte de los padres.

Para ello, desde la aplicación móvil, la interfaz está diseñada en un modelo de pestañas, con iconos claros y que representan las diferentes secciones de la aplicación. Se informa de manera activa y retroactiva al usuario cuando accede al contenido de los mensajes y de cuándo envía respuestas a los mismos y los padres solo deberán introducir su número de teléfono para iniciar sesión y su nombre y apellidos si nos encontráramos en el registro. La asignación de padres/alumnos se realizaría por parte del centro.

Respecto a la versión anterior, se han decidido modificar diferentes cuestiones, tanto en la parte del *backend* como en la aplicación móvil, para mejorar tanto la seguridad como la usabilidad, así como la implementación de nuevas funcionalidades, para conseguir una comunicación entre padres y profesores más eficiente.

# 2. Competencias específicas cubiertas

## 2.1 Competencias Generales

### G2

*Aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posean las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio.*

Dado que el proyecto parte de una primera fase y que en esta fase trabajamos dos alumnos para su desarrollo, a la hora de aplicar cambios en el sistema existente y desarrollar los nuevos, hemos tenido que justificar cada decisión tomada.

### G4

*Transmitir información, ideas, problemas y soluciones a un público tanto especializado como no especializado.*

Durante el desarrollo del proyecto hemos explicado nuestros objetivos, tanto a gente especializada cómo a personas que potencialmente podría usar nuestra aplicación para, de esta forma, poder obtener más información y conseguir un mejor resultado.

## 2.2 Comunes a la Ingeniería Informática

### CII01

*Capacidad para diseñar, desarrollar, seleccionar y evaluar aplicaciones y sistemas informáticos, asegurando su fiabilidad, seguridad y calidad, conforme a principios éticos y a la legislación y normativa vigente.*

Durante el proyecto hemos tenido que diseñar y desarrollar la aplicación móvil y *API* de conexión con el *backend* garantizando el control de a qué datos se puede acceder en cada momento y la seguridad del sistema.

### CII03

*Capacidad para comprender la importancia de la negociación, los hábitos de trabajo efectivos, el liderazgo y las habilidades de comunicación en todos los entornos de desarrollo de software.*

Debido al carácter colaborativo del proyecto, ha sido de vital importancia establecer una buena comunicación entre los dos autores de las dos partes que componen el mismo, así como con los miembros que trabajaron en la primera iteración del proyecto. Por ello, sobre todo en las fases de análisis y diseño, hubo una colaboración constante con el autor del Trabajo de Fin de Grado encargado del *backend*, para así poder evitar, en gran medida, posibles problemas que pudieran surgir posteriores al desarrollo en la integración de ambas aplicaciones, así como a los autores de los Trabajos Fin de Grado anteriores que trabajaron en el proyecto, para conocer hasta dónde se pudo llegar y analizar mejor el estado en el que se encontraba el proyecto anteriormente.

### CII08

*Capacidad para analizar, diseñar, construir y mantener aplicaciones de forma robusta, segura y eficiente, eligiendo el paradigma y los lenguajes de programación más adecuados.*

Para poder desarrollar las mejoras del sistema existente y decidir migrarlo a una nueva tecnología, hemos necesitado analizar, tanto las tecnologías existentes para mejorar el sistema, como realizar un análisis, diseño e implementación de las mejoras a incluir en esta nueva fase.

### CII012

*Conocimiento y aplicación de las características, funcionalidades y estructura de las bases de datos, que permitan su adecuado uso, y el diseño y el análisis e implementación de aplicaciones basadas en ellos.*

Debido a las dimensiones del proyecto, nos hemos visto obligados a realizar un análisis exhaustivo y un diseño preciso de la base de datos del sistema, para poder implementar las mejoras para esta segunda fase.

### CII013

*Conocimiento y aplicación de las herramientas necesarias para el almacenamiento, procesamiento y acceso a los Sistemas de información, incluidos los basados en web.*

Dado que el proyecto completo consta de dos aplicaciones y un portal web para la gestión por parte del centro, como ya se ha mencionado previamente, se necesita desarrollar una API que nos permita comunicar ambas aplicaciones sobre el mismo sistema de base de datos.

### CII017

*Capacidad para diseñar y evaluar interfaces persona computador que garanticen la accesibilidad y usabilidad a los sistemas, servicios y aplicaciones informáticas.*

Por el formato móvil de nuestra aplicación, resultó muy importante identificar los elementos a destacar dentro de nuestra interfaz, simplificándola lo mayor posible para conseguir una destacable facilidad de uso y sencillez.

### CII018

*Conocimiento de la normativa y la regulación de la informática en los ámbitos nacional, europeo e internacional.*

Hemos realizado una búsqueda concisa en referencia a aquellos apartados de la legislación actual que se aplican en los distintos apartados de nuestra aplicación, para cumplir de manera adecuada con la normativa y regulación vigentes.

## 2.3 Tecnologías de Información (TI)

### TI06

*Capacidad de concebir sistemas, aplicaciones y servicios basados en tecnologías de red, incluyendo Internet, web, comercio electrónico, multimedia, servicios interactivos y computación móvil.*

Debido a la naturaleza de nuestro proyecto, se ha concebido un sistema que integra una aplicación móvil para la gestión del día a día de padres y profesores además de una plataforma web para la gestión por parte del centro.

### TI07

*Capacidad para comprender, aplicar y gestionar la garantía y seguridad de los sistemas informáticos.*

Debido a la necesidad de manejo de datos personales, se ha hecho necesario aplicar medidas de seguridad en nuestro proyecto para garantizar que solo accedan a los datos las personas que deben hacerlo.

## 2.4 Ingeniería del Software (IS)

### IS03

*Capacidad de dar solución a problemas de integración en función de las estrategias, estándares y tecnologías disponibles.*

Haciendo uso del sistema de control de versiones (*Git*), hemos podido trabajar los dos alumnos, cada uno en su respectiva aplicación, de manera independiente, sin interrumpir el trabajo del otro a pesar de tener una parte común en el *backend*. Así como, analizar el trabajo realizado en la fase anterior.

## 2.5 Trabajo Fin de Grado

### TFG01

*Ejercicio original a realizar individualmente y presentar y defender ante un tribunal universitario, consistente en un proyecto en el ámbito de las tecnologías específicas de la Ingeniería en Informática de naturaleza profesional en el que se sinteticen e integren las competencias adquiridas en las enseñanzas.*

Teniendo en cuenta que es un proyecto complejo y que hemos tenido que analizar el estado actual del proyecto y diseñar las nuevas funcionalidades, tuvimos que pasar por todas las fases de un desarrollo de proyecto completo lo que nos ha permitido emplear muchos de los conocimientos adquiridos en el grado.

# 3. Aportaciones

## 3.1 Entorno socioeconómico

Este proyecto, como en su primera fase, persigue el objetivo básico de mejorar la comunicación directa entre centro/profesores y padres. Además, intenta solucionar otra gran problemática actual de la comunicación entre centros y padres, el uso del papel.

El uso del papel para el tratamiento de las circulares informativas, así como para la petición de reuniones de tutorías y recogida de autorizaciones para la realización de actividades, conlleva a diferentes problemas. Estos problemas abarcan desde el nivel de gestión y optimización para el centro hasta temas relacionados con el impacto en el medio ambiente. Es necesario reconocer que uno de los problemas que más afectan a nuestro ambiente hoy en día es la falta de conciencia en el uso del papel, ya que su uso desmedido genera grandes cantidades de basura. Debemos tener muy en cuenta también que el proceso de fabricación del papel es otro problema por la utilización de productos contaminantes como el cloro (que es utilizado para el blanqueo del papel).

Además, no todas las empresas papeleras obtienen la materia prima de las llamadas "granjas de árboles", sino que recurren a la tala ilegal causando así una gran pérdida para el planeta.

Por otro lado, para el centro y los profesores conlleva multitud de beneficios ya que, en una herramienta, pueden centralizar las medidas de comunicación con los padres. Además, obtiene múltiples ventajas como el poder tener copias de seguridad del sistema, que, con el sistema tradicional de papel, es más costoso debido a que conllevaría a duplicados de los documentos. A nivel de gestión, este proyecto provee al centro y docentes de una herramienta en la que gestionar las comunicaciones con los padres. Todo ello disponible desde un dispositivo móvil o un ordenador.

## 3.2 Personal

A nivel personal, el desarrollo de este proyecto ha sido una gran experiencia. Al partir de un proyecto ya iniciado y tener que analizarlo, realizar propuestas de mejoras, fortalecer sus puntos débiles frente a los competidores, me ha ayudado a darme cuenta del potencial alcanzado estos años en el transcurso de la carrera adquiriendo las habilidades necesarias para poder realizar el trabajo de un graduado en Ingeniería Informática. Todo ello me ha permitido trabajar en las fases del análisis, diseño e implementación de un proyecto.

Tener que remodelar la estructura ya definida para incorporar mejoras y nuevas funcionalidades y tener que estudiar las tecnologías utilizadas y nuevas alternativas para seleccionar la mejor opción para el desarrollo, ha sido una gran oportunidad para poner en práctica conocimientos de múltiples asignaturas de la carrera. Con dichas asignaturas

individualmente parece que no alcanzas las competencias necesarias, pero, en su conjunto, me han dotado de capacidades necesarias para la toma de decisiones con criterio para poder desarrollar un proyecto tan ambicioso como el que se trata.

Además, el tener que trabajar mano a mano con otro compañero en el análisis de la aplicación, así como tener que tratar con un proyecto ya empezado, me ha permitido mejorar mis capacidades de organización a la hora de realizar trabajo en equipo. Esto es debido a que, en ciertos puntos del trabajo, aunque cada uno realizaba sus apartados individualmente, el desempeño de uno dependía de lo realizado por el otro. Además, gracias al trabajo realizado en la fase de análisis y diseño previos, hemos estructurado las fases de desarrollo posteriores para que el trabajo fuese fluido, enseñándome la importancia de estas fases para cualquier tipo de desarrollo.

# 4. Metodología de trabajo, planificación del proyecto y pre-análisis

En el desarrollo de software, una metodología hace cierto énfasis en el entorno en el cual se plantea y estructura el desarrollo de un sistema. Una metodología de desarrollo de software, consiste principalmente en hacer uso de diversas herramientas, técnicas, métodos y modelos para el desarrollo.

## 4.1 Metodología de trabajo.

La [21] metodología de trabajo que hemos utilizado se basa en el modelo de ciclo de vida del software basado en prototipos. Esta metodología pertenece a los modelos de desarrollo evolutivo, es decir, permite desarrollar versiones de la aplicación que son cada vez más completas y complejas, hasta llegar al objetivo final deseado. Puesto que partimos de un proyecto inicial, el cual debe ser migrado a una nueva tecnología y añadir nuevas funcionalidades, este lo podemos tomar como producto final de una de las etapas, detectando los cambios necesarios en el proyecto actual, y desarrollando los nuevos prototipos con los nuevos requisitos del sistema. Esto nos ha permitido averiguar cuáles son las funcionalidades que se debían mejorar y cuales se tenían que añadir.

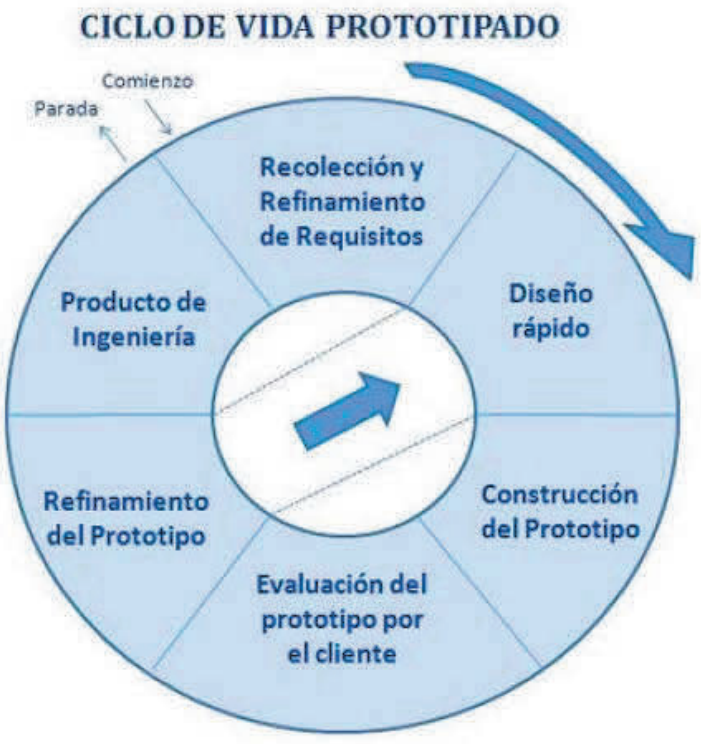


*Ilustración 1: Metodología basada en prototipos.*

Como podemos observar en la *Ilustración 1*, la metodología basada en prototipos consiste en construir prototipos de forma iterativa, enseñárselos a los clientes para validar las funcionalidades implementadas y obtener retroalimentación. El fin es obtener información para la siguiente iteración, como puede ser modificar alguna funcionalidad ya implementada, nuevas funcionalidades que se deben añadir a la aplicación o funcionalidades que debemos eliminar.

## 4.2 Planificación del proyecto.

Para el desarrollo del proyecto se realizó una planificación inicial con la estimación de las horas que nos llevaría cada tarea. Esta estimación la podemos observar en la *Tabla 1:*

| ***Fases*** | ***Duración Estimada (horas)*** | ***Tareas*** <br> ***(nombre y descripción, obligatorio al menos una por fase)*** |
|---|---|---|
| Estudio previo / Análisis | 50 | Tarea 1.1: Estudio de las nuevas funcionalidades |
| | | Tarea 1.2: Captura de requisitos |
| | | Tarea 1.3: Prototipo de la interfaz de usuario del sistema |
| Diseño / Desarrollo / Implementación | 170 | Tarea 2.1: Diseño de la estructura del sistema |
| | | Tarea 2.2: Diseño de la actualización de la base de datos |
| | | Tarea 2.3: Aprendizaje de los conocimientos necesarios para el desarrollo: Ionic v3, Cordova, Node.js |
| | | Tarea 2.4: Implementación de prototipos de la aplicación móvil |
| Evaluación / Validación / Prueba | 30 | Tarea 3.1: Pruebas de funcionamiento de la aplicación móvil |
| | | Tarea 3.2: Pruebas de interacción entre la aplicación móvil y la aplicación web |

| | | |
|---|---|---|
| | | Tarea 3.3: Pruebas de integración y rendimiento de la aplicación móvil |
| Documentación / Presentación | 50 | Tarea 4.1: Realización de la memoria |
| | | Tarea 4.2: Realización de la presentación del TFG |

*Tabla 1: Plan de trabajo*

Esta aproximación inicial, dista un poco de las horas aplicadas a cada una de las tareas. Dado que partíamos de una primera fase del proyecto, estimamos que la fase de análisis sería algo inferior de lo normal, pero dado la complejidad que contiene la gestión de las citas, este apartado requirió de un poco más de tiempo del estimado. El tratamiento con alguna de las librerías utilizadas, complico un poco también la fase de desarrollo, aunque en esta fue mucho menor el impacto en horas.

## 4.3 Pre-análisis.

Para comenzar con el proyecto, lo primero fue realizar un análisis del estado del arte, así como del estado actual del proyecto, y de las propuestas de trabajo futuros realizadas en la primera fase del mismo. Primero, identificamos cuáles eran las nuevas funcionalidades a incorporar y que aspectos se debían mejorar del estado en el que se había quedado después del primer desarrollo.

Lo que se detectó en primer lugar fue la necesidad de migración de la tecnología usada en el anterior proyecto a [22] *Ionic 3*, ya que ofrece muchas más ventajas, aunque ello conllevara desarrollar nuevamente las funcionalidades existentes. Esto es debido a que en estas funcionalidades se detectaron mejoras a realizar. Además, se vio la necesidad de implementación de un sistema seguro de consultas a la [23][24] *API REST*, ya que en el estado en el que se encuentra se puede realizar peticiones de cualquier dato sin ningún tipo de control, por lo que migrar todo el sistema actual cobra mayor sentido.

Una vez analizado el estado actual del proyecto y del arte y seleccionadas las nuevas funcionalidades a incorporar en el sistema, identificamos las modificaciones que aparecieron en los actores actuales del sistema (Administrador y padre/madre/tutor), así como la incorporación de un nuevo actor, el profesor. Para ello, hicimos uso de los diagramas de casos de uso y la especificación de los casos de uso, incorporando los nuevos, pero teniendo en cuenta los actuales, dado que se deben volver a desarrollar, analizándolos nuevamente por si requieren de alguna mejora.

Una vez estructurado el trabajo a implementar en el proyecto, se comenzó con el prototipado. Ello nos permitió afinar las funcionalidades importantes, e incorporar algunas que no parecieron importantes en un primer momento y faltaban por incorporar al desarrollo de la primera fase.

Como se mencionó anteriormente, detectamos la necesidad de migrar el proyecto a una nueva tecnología, dado que la usada actualmente ya está muy anticuada en el panorama

actual y disponemos de mejores herramientas de las cuales hemos hecho uso para el desarrollo.

Lo primero fue comenzar a familiarizarse con las herramientas seleccionadas, así como con el sistema existente. Una vez adquirido el conocimiento necesario sobre las herramientas, se comenzó con el desarrollo de la aplicación, así como a aplicar las mejoras de la *API REST* y desarrollo de las nuevas funcionalidades.

Siguiendo la metodología seleccionada, en el desarrollo del proyecto seguimos una serie de iteraciones detalladas a continuación:

- Iteración 1: En esta iteración se comenzó con la instalación básica del proyecto y con el desarrollo de las funcionalidades existentes en el anterior sistema, utilizando la API REST existente anteriormente.
- Iteración 2: En esta iteración se comenzó con el desarrollo de nuevas funcionalidades y se comenzó con la incorporación del nuevo actor, el profesor. Se creó su sistema de inicio de sesión ya que se diferencia del sistema usado por los padres y se inició la gestión de horarios de tutorías. Además, se comenzó con el desarrollo de la nueva API REST.
- Iteración 3: En esta iteración se desarrollaron las funcionalidades de las citas tanto en los padres como en el profesor.
- Iteración 4: Por último, en esta iteración, se añadieron mejoras en el sistema de mensajes por parte del padre, el cual permite saber cuándo se ha interactuado con alguno de ellos sin necesidad de entrar en el detalle, así como mejoras en el profesor a la hora de gestionar las solicitudes de citas.

# 5. Tecnologías y herramientas utilizadas

## 5.1 Tecnologías

- *Ionic* 3: *Ionic* es un *Software Development Kit (SDK)* de código abierto para el desarrollo de aplicaciones híbridas. Éste se construye sobre *Angular* y *Apache Cordova*. La primera versión fue lanzada en 2013 y, al igual que Angular, sufrió una transformación significativa en su segunda versión. La gran ventaja de las aplicaciones híbridas es que el código sólo se realiza una vez y sirve tanto para *iOS* como para *Android*. Cabe destacar que posee una documentación sobresaliente, debido a la cantidad y la calidad de los ejemplos y explicaciones.

- *NPM*: *NPM* es el gestor de paquetes y dependencias de *NodeJS*, el cual se encarga de la instalación de nuevos módulos o librerías que se quieran incluir en el proyecto. Nosotros sólo tenemos que indicar que queremos incluir y él se encarga del resto, ya que, en algunos casos, un módulo puede requerir de unos cuantos más. Tarea que, realizada manualmente, puede ser muy engorrosa.

- *Angular 5*: Según la [25] documentación oficial, *Angular* se define como una plataforma que facilita el desarrollo de aplicaciones web. *Angular* empezó como un *framework MVC* de *JavaScript* en su primera versión de salida en 2010. Desde ese entonces se ha mantenido por *Google* y el código fuente se encuentra disponible en *GitHub*. En 2016 Angular pasó de la versión 1 a la 2 y eso no sólo implicó un cambio de número. El código del *framework* se rescribió desde cero y se pasó a utilizar *TypeScript* en vez de JavaScript. Además, se dejó de lado la arquitectura *MVC* para pasar a una arquitectura basada en componentes, donde la lógica, la vista y el estilo se encapsulan en componentes. Este *framework* suele utilizarse en las llamadas *SPA (single page application*).

- *Apache Cordova*: es un entorno de desarrollo de aplicaciones móviles que permite a los programadores construir aplicaciones para dispositivos móviles haciendo uso de *CSS3*, *HTML5* y *Javascript* en lugar de lidiar con las *APIs* específicas de cada plataforma, así como encapsular *CSS*, *HTML* y código *Javascript* dependiendo de la plataforma del dispositivo.

- *MySQL*: es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual *GPL*/Licencia comercial por *Oracle Corporation* y está considerada como la base datos de código abierto más popular del mundo, sobre todo para entornos de desarrollo web.

- *Symfony 3*: es un *framework* diseñado para optimizar el desarrollo de las aplicaciones web basado en el patrón Modelo-Vista-Controlador (*MVC*). Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de

desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación.

- *PHP 7*: es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los 20 primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento *HTML* en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de *PHP* que genera la página web resultante. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

- *Doctrine*: es un mapeador de objetos-relacional (*ORM*) escrito en *PHP* que proporciona una capa de persistencia para objetos *PHP*. Es una capa de abstracción que se sitúa justo encima de un *SGBD* (sistema de gestión de bases de datos).

- *TypeScript*: es un lenguaje de programación de código abierto desarrollado por *Microsoft*, el cual cuenta con herramientas de programación orientada a objetos, muy favorable si se tienen proyectos grandes. Anders Hejlsberg, arquitecto principal del desarrollo del lenguaje de programación *C#*, es el principal participante en el desarrollo de este lenguaje. *TypeScript* convierte su código en *Javascript* común. Es llamado también *Superset* de *Javascript*, lo que significa que, si el navegador está basado en *Javascript*, este nunca llegará a saber que el código original fue realizado con *TypeScript* y ejecutará el *Javascript* como lenguaje original.

- *CSS*: es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado. Es muy usado para establecer el diseño visual de las páginas web, e interfaces de usuario escritas en *HTML*.

- *HTML*: es un lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código *HTML*) para la definición de contenido de una página web. Se considera el lenguaje web más importante siendo su invención crucial en la aparición, desarrollo y expansión de la *World Wide Web* (*WWW*). Es el estándar que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales han adoptado.

- *Git*: es un software de control de versiones, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. Gestiona eficientemente el desarrollo distribuido, ofreciendo a cada programador una copia local del

historial de desarrollo. Además, permite que los almacenes de información se puedan publicar vía *HTTP*, *FTP*...

## 5.2 Herramientas

- WebStorm 2018.1: es un IDE para desarrollar en HTML, JavaScript, CSS, SASS, *TypeScript* entre otros, desarrollado en la plataforma JetBrains IntelliJ IDEA. Cuenta con un editor para las tecnologías mencionadas anteriormente, un depurador y refactorización automática para código HTML, JavaScript, CSS, SASS, *TypeScript* y autocompletado de código.

- *PHPStorm* 2018.1: es un *IDE* para desarrollar en *PHP*, aunque también está preparado para desarrollar en lenguajes como *HTML* y *JavaScript*. Esta herramienta está desarrollada en la plataforma *JetBrains IntelliJ IDEA*. Cuenta con un editor para *PHP*, *HTML* y *JavaScript*, un depurador y refactorización automática para código *PHP* y *JavaScript* y autocompletado de código.

- *MySQL Workbench*: es una herramienta visual de diseño de bases de datos que integra desarrollo de software, administración de bases de datos, diseño de bases de datos, creación y mantenimiento para el sistema de base de datos *MySQL*.

- *Google Chrome*: navegador web. Cuenta con herramientas de desarrollo para el desarrollo y depurado de dispositivos *Android*.

- *GitHub*: es una plataforma de desarrollo colaborativo de software utilizada para alojar proyectos utilizando el sistema de control de versiones *Git*.

- *StarUML*: es una herramienta que soporta diagramas *UML*.

## 5.3 Servidores

- *XAMPP*: es un paquete de instalación independiente de plataforma, software libre, que consiste principalmente en el sistema de gestión de bases de datos *MySQL*, el servidor web *Apache* que nos aporta un sitio virtual donde poder alojar nuestro proyecto y ver los cambios en tiempo real que vamos realizando en el desarrollo. Además, cuenta con los intérpretes para lenguajes de script *PHP* y *Perl*.

# 6. Normativa y legislación

## 6.1 Licencia de Software

Una [10] licencia de software es un contrato entre el licenciante (autor/titular de los derechos de explotación/distribución) y el licenciatario (usuario consumidor, profesional o empresa) del programa informático, para utilizarlo cumpliendo una serie de términos y condiciones establecidas dentro de sus cláusulas. Es decir, es un conjunto de permisos que un desarrollador le puede otorgar a un usuario en los que tiene la posibilidad de distribuir, usar o modificar el producto bajo una licencia determinada. Además, se suelen definir los plazos de duración, el territorio donde se aplica la licencia (ya que la licencia se soporta en las leyes particulares de cada país o región), entre otros.

Las licencias de software pueden establecer entre otras cosas: la cesión de determinados derechos del propietario al usuario final sobre una o varias copias del programa informático, los límites en la responsabilidad por fallos, el plazo de cesión de los derechos, el ámbito geográfico de validez del contrato e incluso pueden establecer determinados compromisos del usuario final hacia el propietario, tales como la no cesión del programa a terceros o la no reinstalación del programa en equipos distintos al que se instaló originalmente.

### 6.1.1 Licencia pública general de GNU

La [11] licencia pública general de *GNU* es una licencia de derecho de autor ampliamente usada en el mundo del software libre y código abierto y garantiza a los usuarios finales (personas, organizaciones, compañías) la libertad de usar, estudiar, compartir (copiar) y modificar el software. Su propósito es doble: declarar que el software cubierto por esta licencia es libre y protegerlo (mediante una práctica conocida como *copyleft*) de intentos de apropiación que restrinjan esas libertades a nuevos usuarios cada vez que la obra es distribuida, modificada o ampliada.

Esta licencia es utilizada por: *Git*, *MySQL*, *XAMPP* y *Doctrine*.

[12] *XAMPP* hace uso de otras licencias, dado que esta constituidos por otros componentes, los cuales tienen sus propias licencias. Por tanto, habría que estudiar cada una de las licencias de dichos componentes para saber cuáles son los límites de esta herramienta en su conjunto.

### 6.1.2 Licencia comercial

La [13] licencia comercial se conoce como software propietario o, dicho correctamente, privativo. Este tipo de licencia tiene como finalidad omitir la posibilidad acceder de forma libre a su código fuente, el cual solo se encuentra a disposición de su desarrollador y no se permite su libre modificación, adaptación o incluso lectura por parte de terceros.

Esta licencia es utilizada por: *MySQL* y *StarUML*.

*MySQL* es software de fuente abierta. Fuente abierta significa que cualquier persona tiene la posibilidad de usarlo y modificarlo. Cualquiera puede bajar el código fuente de *MySQL* y usarlo sin pagar. Los interesados pueden estudiar el código fuente y ajustarlo a sus necesidades. *MySQL* usa el *GPL* (*GNU General Public License*) para definir qué puede hacer y qué no con el software en diferentes situaciones. En caso de no ajustarse a la licencia *GPL* o si el usuario requiere introducir código *MySQL* en aplicaciones comerciales, puede comprar una versión comercial licenciada.

### 6.1.3 Software gratis

[14] Software gratis se define en un tipo de software que se distribuye sin costo, disponible para su uso, pero que mantiene el copyright. Por ello, no se puede modificar o utilizar libremente como ocurre con el software libre.

Esta licencia es utilizada por: *Google Chrome*.

### 6.1.4 *MIT*

Es una licencia permisiva lo que significa que impone muy pocas limitaciones en la reutilización y por tanto posee una excelente compatibilidad de licencia. La [15] licencia *MIT* permite reutilizar software dentro de software propietario. Por otro lado, la licencia *MIT* es compatible con muchas licencias *copyleft*, como de *GNU*.

Esta licencia es utilizada por: *Ionic*, *Angular* y *Symfony*.

### 6.1.5 Licencia PHP

La [16] licencia *PHP* es la licencia bajo la que se publica el lenguaje de programación PHP. De acuerdo a la *Free Software Foundation* es una licencia de software libre no *copyleft* y una licencia de código abierto según la *Open SourceInitiative*. Debido a la restricción en el uso del término "*PHP*", no es compatible con la licencia *GPL*.

### 6.1.6 Licencia *Apache*

La [17] licencia *Apache* es una licencia de software libre permisiva creada por la *Apache Software Foundation* (ASF). La licencia *Apache* requiere la conservación del aviso de derecho de autor y el descargo de responsabilidad, pero no es una licencia *copyleft* ya que no requiere la redistribución del código fuente cuando se distribuyen versiones modificadas.

Esta licencia es utilizada por: *Apache Cordova*.

### 6.1.7 Licencia para educación de *JetBrains*

La [18] licencia de *JetBrains* para educación tiene un formato de suscripción anual y es gratuita para los miembros de la comunidad universitaria. No obstante, comprende una serie de restricciones en lo que respecta a su uso, por lo que no se permite:

- Alquilar, reproducir, modificar, adaptar, crear trabajos derivados, vender, sublicenciar o transferir los productos, o proveer dicho producto asociado a nuestra cuenta a terceros.
- El uso de ingeniería inversa, descompilar, desensamblar, modificar, traducir los productos o cualquier intento de descubrir el código fuente de los productos.
- Eliminar u ocultar cualquier aviso de propiedad o de otro tipo contenidos en los productos.
- Usar los productos para propósitos comerciales.

Esta licencia es utilizada por: *WebStorm* y *PhpStorm*.

## 6.2 Seguridad de los Datos

El [19] Reglamento General de Protección de Datos (RGPD) es el nuevo reglamento general de protección de datos que entró en vigor en mayo de 2016, y es de aplicación obligatoria para todas las empresas de la Unión Europea desde del 25 de mayo de 2018, y otorga un mayor control y seguridad a los ciudadanos sobre su información personal en el mundo 2.0. El RGPD amplía sus derechos a decidir cómo desean que sus datos sean tratados y a cómo quieren recibir información de las empresas.

Con la entrada en vigor de dicha normativa, aparecen nuevos elementos a tener en cuenta sobre la vieja [20] Ley Orgánica de Protección de Datos (LOPD).

- Derecho al olvido

  Es el derecho que tienen los ciudadanos a solicitar, y conseguir de los encargados, que los datos personales sean suprimidos cuando estos ya no sean necesarios para el fin para el que fueron reunidos, cuando se haya revocado el consentimiento o cuando estos se hayan obtenido de forma ilegal.

- Derecho a la portabilidad

  Implica que el interesado que haya proporcionado sus datos a un responsable que los esté tratando de forma digitalizada podrá requerir recobrar esos datos en un formato que le permita su traslado a otro responsable.

- Cambios en la obtención del consentimiento

  El Reglamento pide que el consentimiento, con carácter general, sea libre, informado, específico e inequívoco. Las empresas deberán revisar la forma en la que obtienen y guardan el consentimiento. Actualmente existen prácticas que se encuadran en el llamado consentimiento tácito y que son aceptadas con la actual normativa, pero han dejado de serlo desde la entrada en vigor del Reglamento. Para poder considerar que el consentimiento es "incuestionable", el Reglamento requiere que haya una declaración de los interesados o una acción positiva que apunte al acuerdo del interesado. La aceptación no puede deducirse del silencio o de la inacción de los ciudadanos. Se exige que el consentimiento tenga que ser "manifiesto" en determinados casos, como puede ser para autorizar el tratamiento de datos sensibles. Por tanto, el consentimiento tiene que ser verificable y quienes recopilen datos personales deben poder probar que el afectado les concedió su consentimiento.

Teniendo en cuenta los aspectos mencionados anteriormente, para que la aplicación desarrollada cumpla la ley, hay que asegurar la protección de los datos.

Esta aplicación hace uso de datos personales. Estos datos son:

- Nombre y apellidos (Padres, Alumnos y Profesores)
- Teléfono (Padres)

Por tanto, basándonos en estos datos hay que asegurar que los datos no puedan ser accedidos por terceros, para ello se seguirán las siguientes pautas:

- La base de datos estará protegida bajo contraseña y solo el rol que llamaremos "Super Admin" tendrá acceso.
- Los recursos estarán controlados mediante un token por lo que nadie externo a la plataforma podrá acceder a dichos recursos.

- Los datos de los alumnos y los padres se alojarán en la plataforma siempre y cuando los padres den su consentimiento en el centro en el que quieren participar.
- El sistema garantiza que los datos sólo serán alojados en la base de datos durante el tiempo necesario para cumplimentar la funcionalidad que se requiere por parte de los centros. En caso de no ser necesarios en cierto punto, o que el padre/madre/tutor soliciten la baja en el sistema, los datos serán eliminados de la base de datos.
- Los datos que utilizará el sistema serán los estrictamente necesarios para el correcto funcionamiento de este, nunca se pedirá un dato innecesario.

# 7. Análisis

En esta sección de la memoria nos centraremos en la etapa de Análisis de nuestra aplicación, incluyendo apartados del análisis de la fase anterior y cambios respecto a la misma.

## 7.1 Actores

Primeramente, hablaremos de los actores identificados durante el análisis, separando cada uno y explicando que papel juegan en nuestra aplicación. Además de los dos que ya se encontraban en el sistema, ahora surge la necesidad de añadir al profesor.

### 7.1.1 Administrador

El administrador es la persona responsable de gestionar el centro, así como el encargado de la gestión de cursos, registro de profesores, asignación de alumnado al curso y de realizar las distintas gestiones posibles en el menú de mensajería del centro. Por ende, las funciones que podrá realizar se dividen en dos grupos:

- Gestión del alumnado, asignaturas, profesores y cursos del centro:
  Estas funcionalidades son necesarias para poder llevar acabo aquellas que realmente persiguen el objetivo final del sistema. Entre ellas se encuentran:
  - Creación de cursos.
  - Registrar alumnos.
  - Asociar y desasociar alumnos de un curso.
  - Asociar y desasociar padres de un alumno.
  - Registrar profesores en el centro.
  - Asociar el profesor que será el tutor de un curso.
  - Importar automáticamente los datos del centro.
  - Cambiar los alumnos de curso.

- Gestión de la mensajería:
  Estas funcionalidades llevan a cabo el objetivo principal del sistema. Permiten enviar distintos tipos de mensajes a los padres de alumnos, tales como:
  - Circulares generales que contienen información relevante sobre algún aspecto específico del centro.
  - Autorizaciones de actividades para el alumnado, que permiten a los padres firmarlas desde su teléfono móvil. También se permite visualizar un listado de aquellos alumnos que hayan sido autorizados y aquellos que no para una determinada actividad.

- Encuestas para que los padres formen parte de las tomas de decisiones que el centro considere oportunas. Además, se podrán visualizar los resultados de dichas encuestas.

Respecto a la primera fase, lo nuevo que ha surgido es la necesidad de control sobre los profesores y el curso de los que son tutores, permitiendo aplicar el sistema de citas.

### 7.1.2 Padre/Madre/Tutor

En este rol se engloba a los padres/madres/tutores legales del alumnado. Para poder tener acceso al sistema, deben de pasar por el centro para que le realicen la asociación de sus hijos. Dichos actores podrán realizar las siguientes acciones:

- Ver el contenido de los distintos tipos de mensajes que el centro les ha enviado (circulares generales, autorizaciones de actividades y encuestas).
- Filtrar dichos mensajes por su asunto.
- Firmar las autorizaciones, haciendo uso de un código de seguridad establecido por el padre/madre/tutor legal (inicialmente el mensaje SMS recibido en el inicio de sesión).
- Confirmación de lectura de las circulares, respuesta de encuestas y autorizaciones.
- Responder a las encuestas enviadas por el centro.
- Seleccionar los centros en los que tenga hijos.
- Ver y desasociar a sus hijos previamente asociados por el centro.
- Ver calendario de horarios de citas disponibles de los profesores de sus hijos.
- Solicitar citas a los profesores de sus hijos.
- Cancelar una cita.
- Ver citas pendientes.
- Registro en la aplicación.
- Iniciar sesión en el sistema.

Como se puede deducir, este rol tiene un papel principal en nuestra aplicación ya que será el usuario final de la aplicación móvil. Debe incluirse, por ende, una fase de registro en el sistema por parte de los usuarios pertenecientes a este rol, de lo contrario la otra alternativa es que los registrase el centro, pudiéndose dar el caso de que los padres no tuvieran la aplicación instalada y el centro pensara de manera errática que los mensajes están llegando a sus destinatarios cuando no lo estarían haciendo.

Respecto a la fase anterior, se ha añadido todo lo que es la gestión de tutorías con los profesores de sus hijos ya que es uno de los puntos que se busca reforzar en esta segunda fase. También, se ha implementado un sistema de diferenciación de mensajes leídos o sobre los que se ha realizado alguna acción, permitiendo que, a la larga, cuando se nos acumulen múltiples mensajes de cada tipo en el transcurso del año académico,

podamos diferenciar de una forma muy visual los mensajes que aun requieren de una acción.

### 7.1.3 Profesor

El profesor es un nuevo actor respecto a la fase anterior y es quien mantendrá contacto directo con los padres, para que estos puedan tener un mayor control de sus hijos en el centro. Las tareas que podrá realizar el profesor son las siguientes:

- Establecer horarios de tutorías.
- Borrar horarios de tutorías.
- Crear citas con padres de sus alumnos.
- Ver citas pendientes.
- Ver las solicitudes de citas.
- Confirmar peticiones de citas.
- Cancelar citas y peticiones.

## 7.2 Requisitos

Para representar los requisitos funcionales del sistema nos hemos apoyado de diagramas UML (Lenguaje Unificado de Modelado).

La razón de haber escogido UML es que, además de la experiencia que previamente hemos adquirido con este lenguaje de modelado en el transcurso del grado, es el más utilizado en la actualidad en lo que respecta a sistemas software.

UML es un estándar aprobado por la ISO (Organización Internacional de Normalización) por lo que, cualquier diagrama creado con UML puede ser interpretado por cualquier persona que conozca el estándar.

El objetivo de este Trabajo de Fin de Grado solo abarca el desarrollo de la aplicación móvil utilizada por los padres y profesores, por lo que creemos adecuado centrarnos únicamente en los casos de uso de dichos actores.

Los casos de uso relacionados con la administración del centro están explicados en detalle en el Trabajo de Fin de Grado titulado: "*Análisis, Diseño e Implementación de un Backend para la Comunicación entre Centros Educativos y Padres de Alumnos II*".

### 7.2.1 Casos de uso

Dado que partimos de una primera fase del proyecto, se reutilizará parte del análisis realizado en la etapa anterior, remarcando los nuevos casos de uso. Esto es así ya que en la migración del sistema necesitaremos implementar todos los casos de uso y tenerlos definidos. Para diferenciar los anteriores de los nuevos, se cambiará el color de fondo de los nuevos casos de uso definidos, resaltados en naranja.

A continuación, se mostrarán los diagramas de casos de uso para tener una idea más clara de los requisitos funcionales adheridos al sistema, dividido en distintas imágenes como se propuso en la primera fase del proyecto.

#### *7.2.1.1 Padre/Madre/Tutor*

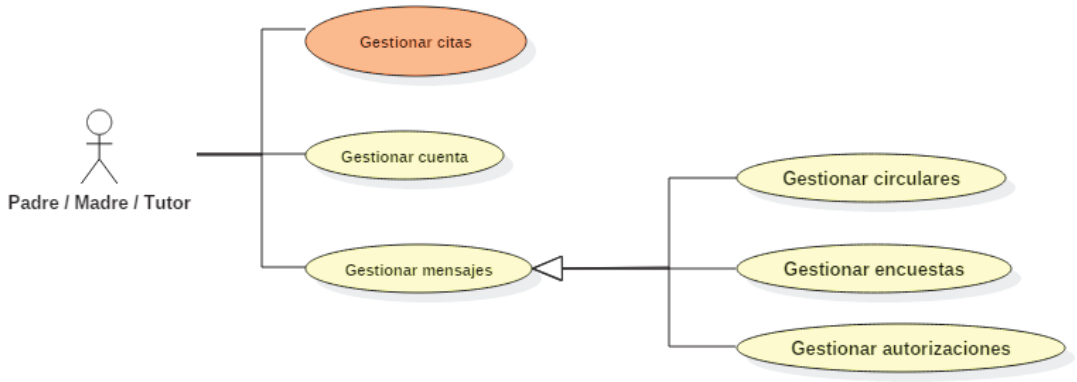


*Ilustración 2: Diagrama de casos de uso (Resumen)*

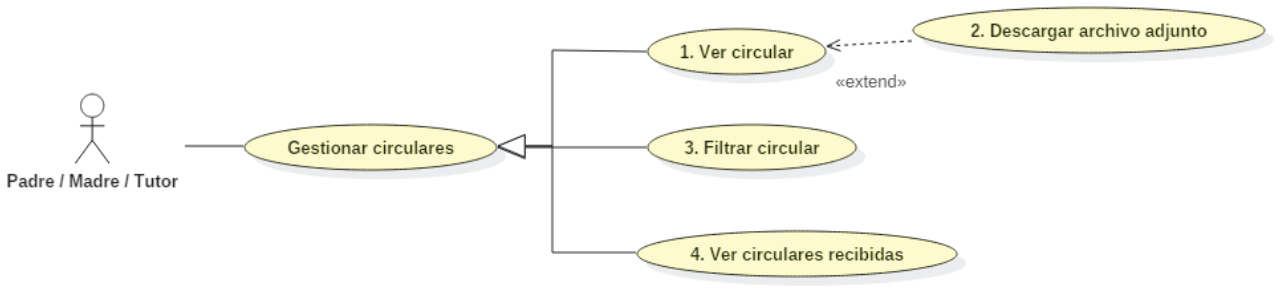


*Ilustración 3: Diagrama de casos de uso (Gestionar circulares)*

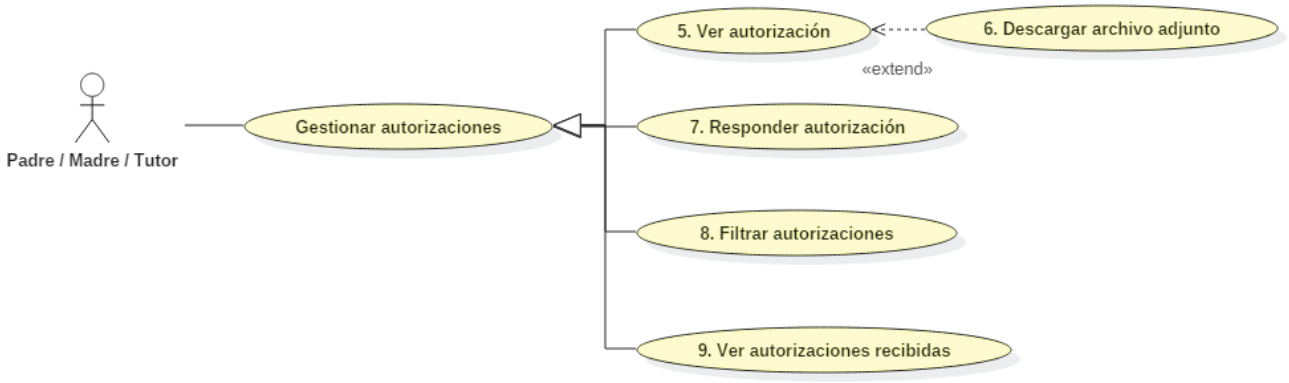


*Ilustración 4: Diagrama de casos de uso (Gestionar autorizaciones)*

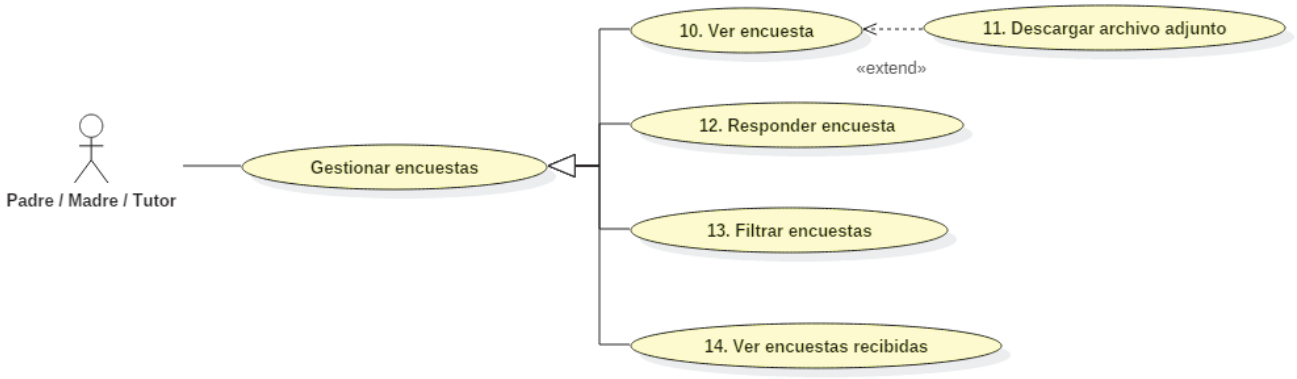


*Ilustración 5: Diagrama de casos de uso (Gestionar encuestas)*

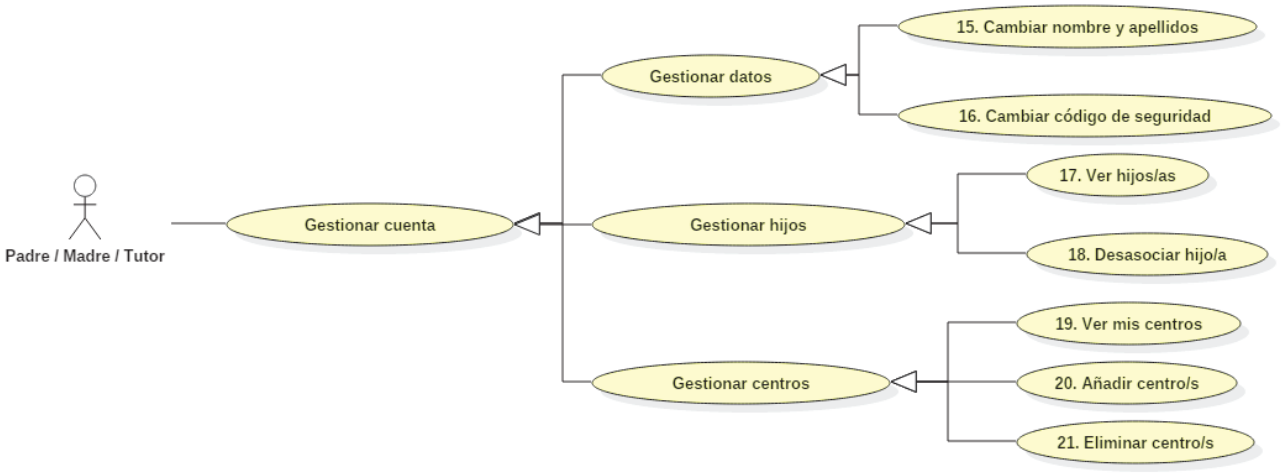


*Ilustración 6: Diagrama de casos de uso (Gestionar cuenta)*

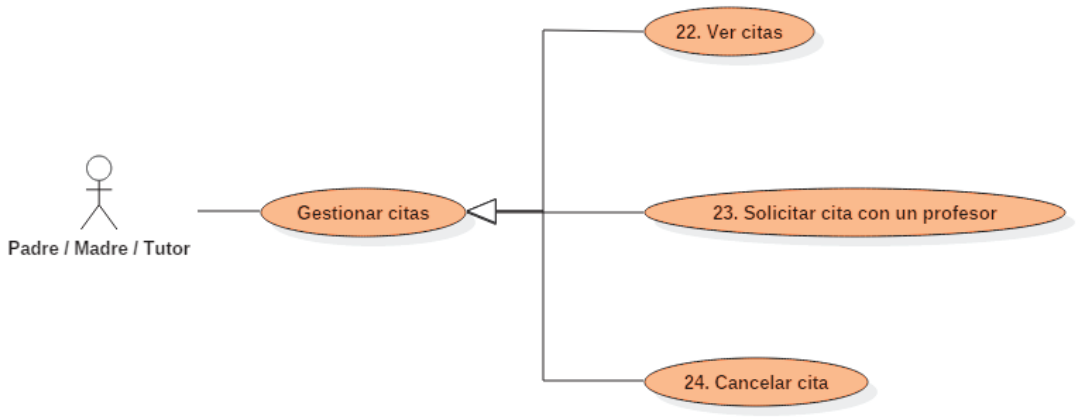


*Ilustración 7: Diagrama de casos de uso (Gestionar citas)*

Bajo estas líneas, especificaremos en la *Tabla 2* cada uno de los casos de uso representados anteriormente en el diagrama, acompañados de una breve descripción de estos.

| ID | Actor | Descripción |
|---|---|---|
| 1 | Padre/Madre/Tutor | Se muestra el contenido de la circular seleccionada. |
| 2 | Padre/Madre/Tutor | Se descarga el documento adjunto a la circular que se está visualizando. |
| 3 | Padre/Madre/Tutor | Se muestra el listado de circulares que coinciden con el filtro introducido. |
| 4 | Padre/Madre/Tutor | Se muestra un listado de todas las circulares destinadas al usuario. |
| 5 | Padre/Madre/Tutor | Se muestra el contenido de la autorización seleccionada. |
| 6 | Padre/Madre/Tutor | Se descarga el documento adjunto a la autorización que se está visualizando. |
| 7 | Padre/Madre/Tutor | Se autoriza o desautoriza al hijo/a de la autorización mostrada a realizar la actividad propuesta por el centro. |
| 8 | Padre/Madre/Tutor | Se muestra el listado de autorizaciones que coinciden con el filtro introducido. |
| 9 | Padre/Madre/Tutor | Se muestra un listado de todas las autorizaciones destinadas al usuario. |
| 10 | Padre/Madre/Tutor | Se muestra el contenido de la encuesta seleccionada. |
| 11 | Padre/Madre/Tutor | Se descarga el documento adjunto a la encuesta que se está visualizando. |
| 12 | Padre/Madre/Tutor | Se responde a la encuesta propuesta por el centro. |
| 13 | Padre/Madre/Tutor | Se muestra el listado de encuestas que coinciden con el filtro introducido. |
| 14 | Padre/Madre/Tutor | Se muestra un listado de todas las encuestas destinadas al usuario. |
| 15 | Padre/Madre/Tutor | Se cambia el Nombre y Apellidos actuales del usuario por el introducido. |

| | | |
|---|---|---|
| 16 | Padre/Madre/Tutor | Se cambia el código de seguridad actual del usuario por el introducido. |
| 17 | Padre/Madre/Tutor | Se muestra un listado de los hijos/as asociados al usuario. |
| 18 | Padre/Madre/Tutor | Se desasocia el hijo/a seleccionado del usuario. |
| 19 | Padre/Madre/Tutor | Se muestra un listado de los centros asociados al usuario. |
| 20 | Padre/Madre/Tutor | Se añade el centro seleccionado a la lista de centros permitidos del usuario. |
| 21 | Padre/Madre/Tutor | Se elimina el centro seleccionado de la lista de centros permitidos del usuario. |
| 22 | Padre/Madre/Tutor | Se ve un listado de las citas pendientes |
| 23 | Padre/Madre/Tutor | Se solicita una cita a un profesor de uno de los hijos. |
| 24 | Padre/Madre/Tutor | Se cancela una cita de las pendientes. |

*Tabla 2: Resumen de casos de uso Padre/Madre/Tutor*

#### *7.2.1.2 Profesor*

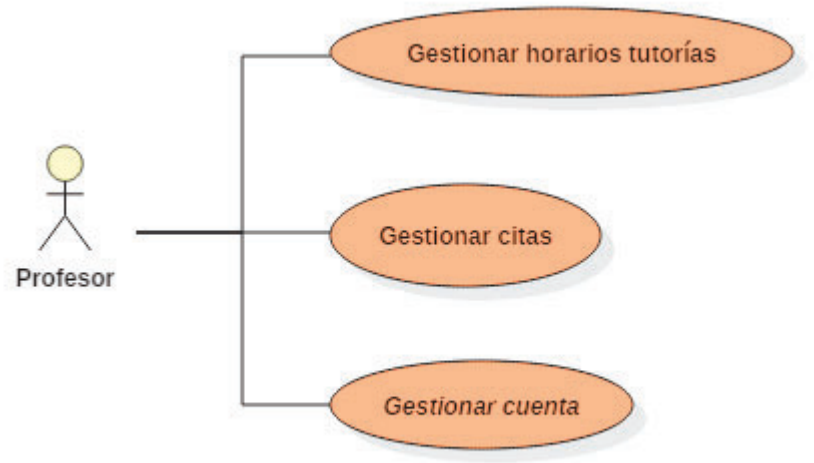


*Ilustración 8: Diagrama de casos de uso (Resumen)*

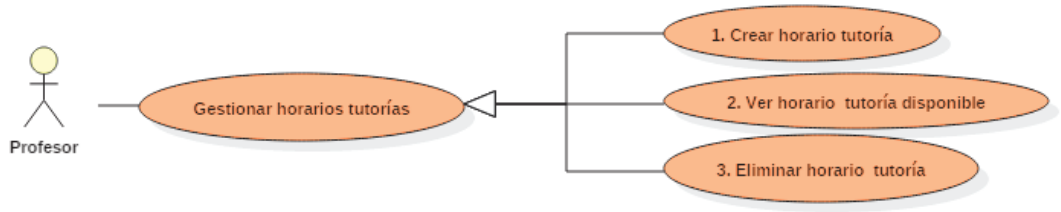


*Ilustración 9: Diagrama de casos de uso (Gestionar horarios tutorías)*

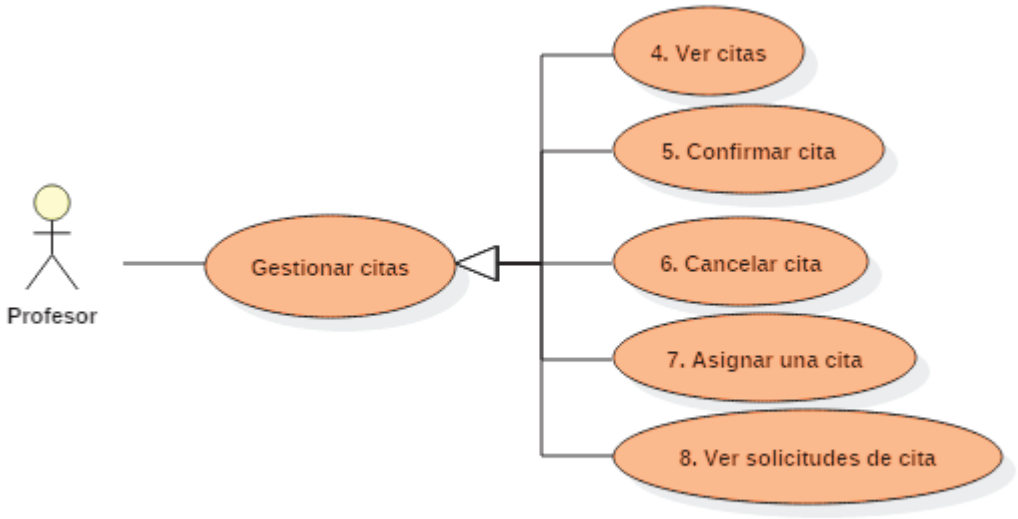


*Ilustración 10: Diagrama de casos de uso (Gestionar citas)*

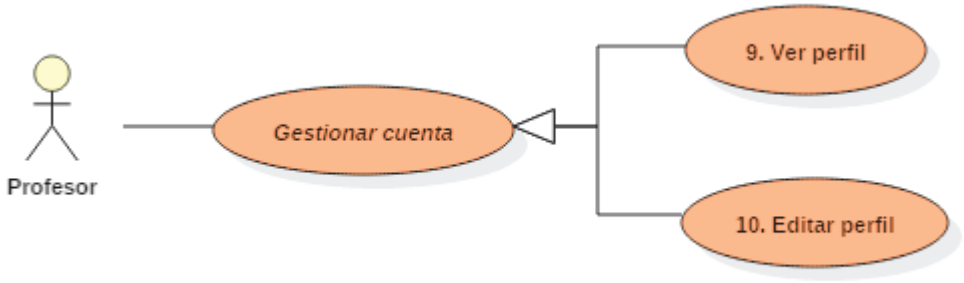


*Ilustración 11: Diagrama de casos de uso (Gestionar cuenta)*

En la *Tabla 3* se muestra una descripción resumida de los casos de uso propios del profesor:

| ID | Actor Principal | Casos de Uso |
|---|---|---|
| 1 | Profesor | Se registra un horario de tutoría. |
| 2 | Profesor | Ver el horario de tutorías disponibles. |
| 3 | Profesor | Se elimina un horario de tutoría del sistema. |
| 4 | Profesor | Se podrá ver las citas que tiene el tutor con un alumno. |
| 5 | Profesor | Se cancela una cita con uno de sus alumnos. |
| 6 | Profesor | Se confirma una solicitud de cita. |
| 7 | Profesor | Se asigna una cita a uno de sus alumnos por parte del profesor. |
| 8 | Profesor | Se muestra un listado con las solicitudes de cita pendientes. |
| 9 | Profesor | Se muestra la información correspondiente al profesor que está conectado en la aplicación |

| | | |
|---|---|---|
| **10** | Profesor | Se modifica la información del profesor (nombre o contraseña). Es necesario que introduzca su contraseña actual, para poder modificarla por una nueva. |

*Tabla 3: Resumen de casos de uso Profesor*

## 7.2.2 Especificación de casos de uso

En este apartado pasaremos a la especificación de los casos de usos que consideramos más importantes para el desarrollo de la aplicación. Puesto que en la fase anterior ya se especificaron algunos, se volverán a nombrar y se añadirán los nuevos casos de uso más relevantes.

### *7.2.2.1 Padre/Madre/Tutor*

| **CASO DE USO** | 22 | **VER CITAS** | |
|---|---|---|---|
| **Descripción** | El padre/madre/tutor ve las citas pendientes con los profesores de sus hijos/as. | | |
| **Actores** | Padre/Madre/Tutor. | | |
| **Precondiciones** | El padre/madre/tutor debe haber iniciado sesión en el sistema. | | |
| **Flujo normal** | **Paso** | **Acción** | |
| | 1 | Se accede al apartado de citas. | |
| | 2 | Se muestra un calendario con las citas y su estado actual. | |
| | 3 | Se selecciona la forma de filtro por día o por mes. | |
| **Postcondiciones** | Se muestra el listado de citas pendientes con el filtro aplicado. | | |
| **Variaciones** | **Paso** | **Acción** | |
| | | | |
| **Extensiones** | **Paso** | **Condición** | **Caso de Uso** |
| | | | |
| **Excepciones** | | | |
| **Observaciones** | | | |

| | |
|---|---|
| | |

*Tabla 4: Especificaciones de casos de uso (Ver citas)*

| **CASO DE USO** | 23 | **SOLICITAR CITA** | |
|---|---|---|---|
| **Descripción** | El padre/madre/tutor solicita una cita con un profesor/a de su hijo/a. | | |
| **Actores** | Padre/Madre/Tutor. | | |
| **Precondiciones** | El padre/madre/tutor debe haber iniciado sesión en el sistema. | | |
| **Flujo normal** | **Paso** | **Acción** | |
| | 1 | Se accede al listado de profesores de sus hijos. | |
| | 2 | Se selecciona el profesor/a con el que se quiere la cita. | |
| | 3 | Se muestra el calendario con las horas disponibles del profesor. | |
| | 4 | Se selecciona el día en el que se quiere solicitar la cita. | |
| | 5 | Se selecciona uno de los horarios disponibles en el que se quiere solicitar la cita. | |
| | 6 | Se realiza la solicitud de cita. | |
| **Postcondiciones** | Se crea una cita con estado pendiente, asociada al profesor/a seleccionado. | | |
| **Variaciones** | **Paso** | **Acción** | |
| | | | |
| **Extensiones** | **Paso** | **Condición** | **Caso de Uso** |
| | | | |
| **Excepciones** | | | |
| **Observaciones** | | | |

*Tabla 5: Especificaciones de casos de uso (Solicitar cita)*

| CASO DE USO | 25 | CANCELAR CITA | |
|---|---|---|---|
| **Descripción** | | El padre/madre/tutor cancela una cita con un profesor/a de uno de sus hijos. | |
| **Actores** | | Padre/Madre/Tutor. | |
| **Precondiciones** | | El padre/madre/tutor debe haber iniciado sesión en el sistema y tener citas asociadas. | |
| **Flujo normal** | **Paso** | **Acción** | |
| | 1 | Se accede al calendario de citas. | |
| | 2 | Se selecciona el día en el que tenemos alguna cita. | |
| | 3 | Se selecciona la cita que se desea cancelar. | |
| | 4 | Se cancela la cita. | |
| **Postcondiciones** | | Se modifica el estado de la cita a cancelada. | |
| **Variaciones** | **Paso** | **Acción** | |
| | | | |
| **Extensiones** | **Paso** | **Condición** | **Caso de Uso** |
| | | | |
| **Excepciones** | | | |
| **Observaciones** | | | |

*Tabla 6: Especificaciones de casos de uso (Cancelar cita)*

### *7.2.2.2 Profesor*

| CASO DE USO | 1 | CREAR HORARIO DE TUTORÍAS |
|---|---|---|
| **Descripción** | | El profesor/a crea el horario deseado para las futuras tutorías con los padres. |
| **Actores** | | Profesor |
| **Precondiciones** | | El profesor debe haber iniciado sesión en el sistema. |

| Flujo normal | Paso | Acción |
|---|---|---|
| | 1 | Se selecciona la fecha de inicio y finalización del rango en el que se quieren añadir horarios de tutorías. |
| | 2 | Se seleccionan los días de la semana en los que se quiere crear el horario para ese rango. |
| | 3 | Se selecciona la hora de inicio, la hora de finalización y el intervalo en el que se creara cada horario. |
| | 4 | Se envía la creación de nuevo horario. |
| **Postcondiciones** | Se añade el nuevo horario a la lista de horarios disponibles. | |
| **Variaciones** | **Paso** | **Acción** |
| | | |
| **Extensiones** | **Paso** | **Condición** | **Caso de Uso** |
| | | | |
| **Excepciones** | | | |
| **Observaciones** | | | |
| | | | |

*Tabla 7: Especificaciones de casos de uso (Crear horario de tutoría)*

| CASO DE USO | 3 | BORRAR HORARIO DE TUTORÍAS |
|---|---|---|
| **Descripción** | El profesor borra parte del horario disponible para las tutorías con los padres. | |
| **Actores** | Profesor | |
| **Precondiciones** | El profesor debe haber iniciado sesión en el sistema. | |
| **Flujo normal** | **Paso** | **Acción** |
| | 1 | Se selecciona la fecha de inicio y finalización del rango en el que se quieren eliminar horarios de tutorías. |
| | 2 | Se acepta la eliminación de horarios. |

| | | | |
|---|---|---|---|
| **Postcondiciones** | Se modifica la lista de horarios disponibles eliminando los del rango seleccionado. | | |
| **Variaciones** | **Paso** | **Acción** | |
| | | | |
| **Extensiones** | **Paso** | **Condición** | **Caso de Uso** |
| | | | |
| **Excepciones** | | | |
| **Observaciones** | | | |
| | | | |

*Tabla 8: Especificaciones de casos de uso (Borrar horario de tutoría)*

| | | | |
|---|---|---|---|
| **CASO DE USO** | 7 | **ASIGNAR UNA CITA** | |
| **Descripción** | El profesor asigna una cita a los padres de uno de sus alumnos. | | |
| **Actores** | Profesor | | |
| **Precondiciones** | El profesor debe haber iniciado sesión en el sistema. | | |
| **Flujo normal** | **Paso** | **Acción** | |
| | 1 | Se accede al calendario del profesor. | |
| | 2 | Se selecciona el día y el horario disponible deseado. | |
| | 3 | Se selecciona el alumno del que se desea solicitar una cita con sus padres. | |
| | 4 | Se asigna el horario al alumno elegido. | |
| **Postcondiciones** | Se añade la nueva cita. | | |
| **Variaciones** | **Paso** | **Acción** | |
| | | | |
| **Extensiones** | **Paso** | **Condición** | **Caso de Uso** |
| | | | |
| **Excepciones** | | | |

| | |
|---|---|
| **Observaciones** | |
| | |

*Tabla 9: Especificaciones de casos de uso (Asignar una cita)*

| **CASO DE USO** | 5 | **CONFIRMAR CITA** | |
|---|---|---|---|
| **Descripción** | | El profesor confirma una solicitud de cita de alguno de los padres de sus alumnos. | |
| **Actores** | | Profesor | |
| **Precondiciones** | | El profesor debe haber iniciado sesión en el sistema y tener solicitudes de citas. | |
| **Flujo normal** | | **Paso** | **Acción** |
| | | 1 | Se accede al listado de solicitudes de citas del profesor. |
| | | 2 | Se selecciona la solicitud que se desea confirmar. |
| | | 3 | Se acepta la petición de cita. |
| **Postcondiciones** | | Se crea cita y se añade a las citas pendientes del profesor. | |
| **Variaciones** | | **Paso** | **Acción** |
| | | | |
| **Extensiones** | | **Paso** | **Condición** | **Caso de Uso** |
| | | | | |
| **Excepciones** | | | |
| **Observaciones** | | | |
| | | | |

*Tabla 10: Especificaciones de casos de uso (Confirmar cita)*

# 8. Diseño

## 8.1 Diseño de la arquitectura del sistema

La programación por capas es un modelo de desarrollo software en el que el objetivo primordial es la separación (desacoplamiento) de las partes que componen un sistema software o también una arquitectura cliente-servidor: lógica de negocios, capa de presentación y capa de datos. De esta forma, por ejemplo, es sencillo y mantenible crear diferentes interfaces sobre un mismo sistema sin requerirse cambio alguno en la capa de datos o lógica.

La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, solo afectará al nivel requerido sin tener que revisar en el código fuente de otros módulos, dado que se habrá reducido el acoplamiento informático hasta una interfaz de paso de mensajes.

Además, permite distribuir el trabajo de creación de una aplicación por niveles; de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles, de forma que basta con conocer la *API* que existe entre niveles.

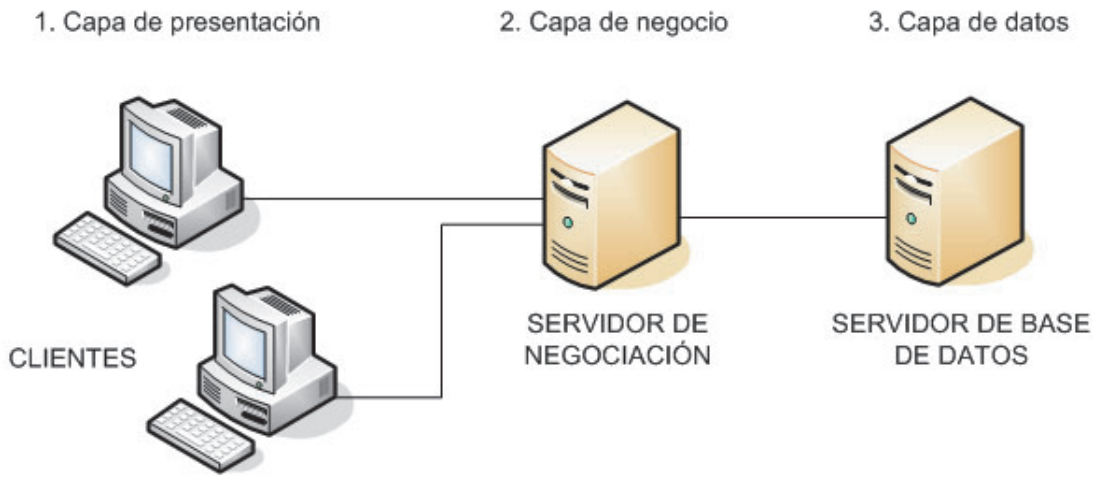


*Ilustración 12: Arquitectura multinivel*

La arquitectura seleccionada como podemos ver en la *Ilustración 12*, nos permite, como ya hemos explicado, tener aislados cada uno de los componentes a desarrollar en el proyecto, como son el cliente web, el cliente móvil, y la *API REST* de la cual se obtendrán los datos. Esta arquitectura además nos permite que sea fácilmente escalable, que su mantenimiento sea más sencillo gracias al desacoplamiento de las partes y mayor seguridad ya que la lógica de negocio y los datos están alojados en un único servidor.

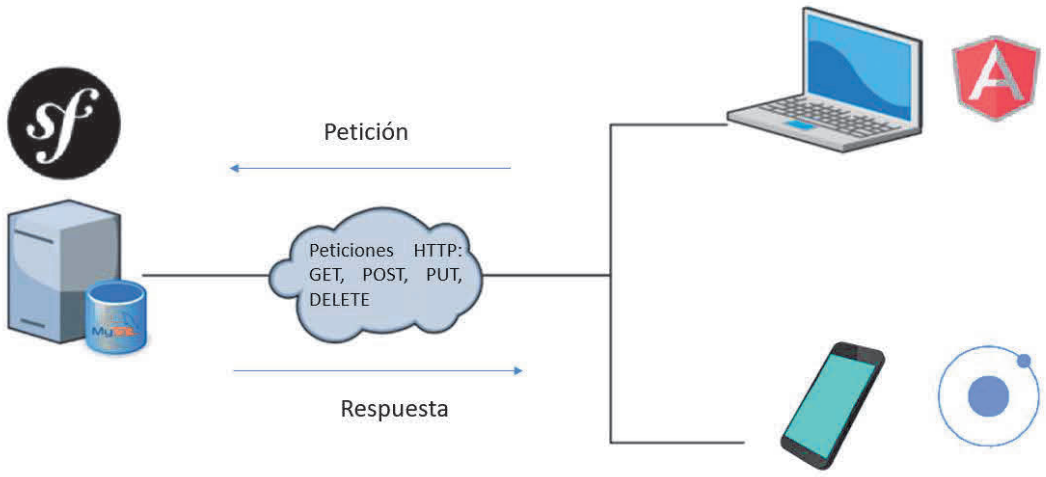


*Ilustración 13: Sistema implementado*

Como podemos observar en la *Ilustración 13*, el sistema que montaremos estará formado por dos clientes, uno web montado en *Angular 6* y otro móvil desarrollado en *Ionic 3* y en la parte del servidor se encuentran la capa de datos y de lógica de negocio desarrollada con una *API REST* en [26] *Symphony 3*.

### 8.1.1 Frontend

En el *frontend* tenemos dos componentes:

- El portal web desarrollado en *Angular*. Este es utilizado por el administrador del centro para gestionar labores propias del centro, como, por ejemplo: enviar todo tipo de mensajes a los padres (circulares, encuestas y autorizaciones), cambiar los alumnos de un curso a otro, registrar profesores, etc.

- La aplicación móvil desarrollada en *Ionic* 3, es la encargada de gestionar la lectura de las circulares, así como la contestación de encuestas y autorizaciones. Además, es responsable de realizar una gestión de los horarios de tutoría del profesor y las citas con el padre/madre/tutor legal.

### 8.1.2 Backend

La *API REST* que da vida a nuestro *backend* ha sido desarrollada con el *framework PHP* denominado *Symfony*, que a su vez hace uso de *PHP 7* y *Doctrine*. Este, se encuentra alojado en un servidor web *Apache* donde también se encuentra alojada la base de datos *MySQL*.

Este *backend* es usado por ambas aplicaciones (portal web y aplicación móvil) de manera que, el portal web y la aplicación móvil, hacen uso de la misma base de datos.

## 8.2 Diseño de la base de datos

En la primera fase del proyecto, se decidió utilizar el sistema de gestión de base de datos MySQL. Por tanto, en esta segunda fase se seguirá utilizado MySQL para gestionar la base de datos, ya que así se puede reutilizar la base de datos de la primera fase.

A continuación, se mostrará la estructura de la base de datos generada en la primera fase del proyecto y las actualizaciones pertinentes (remarcadas con un borde rojo) que se tuvieron que realizar para añadir al sistema las nuevas funcionalidades que se desarrollaron.

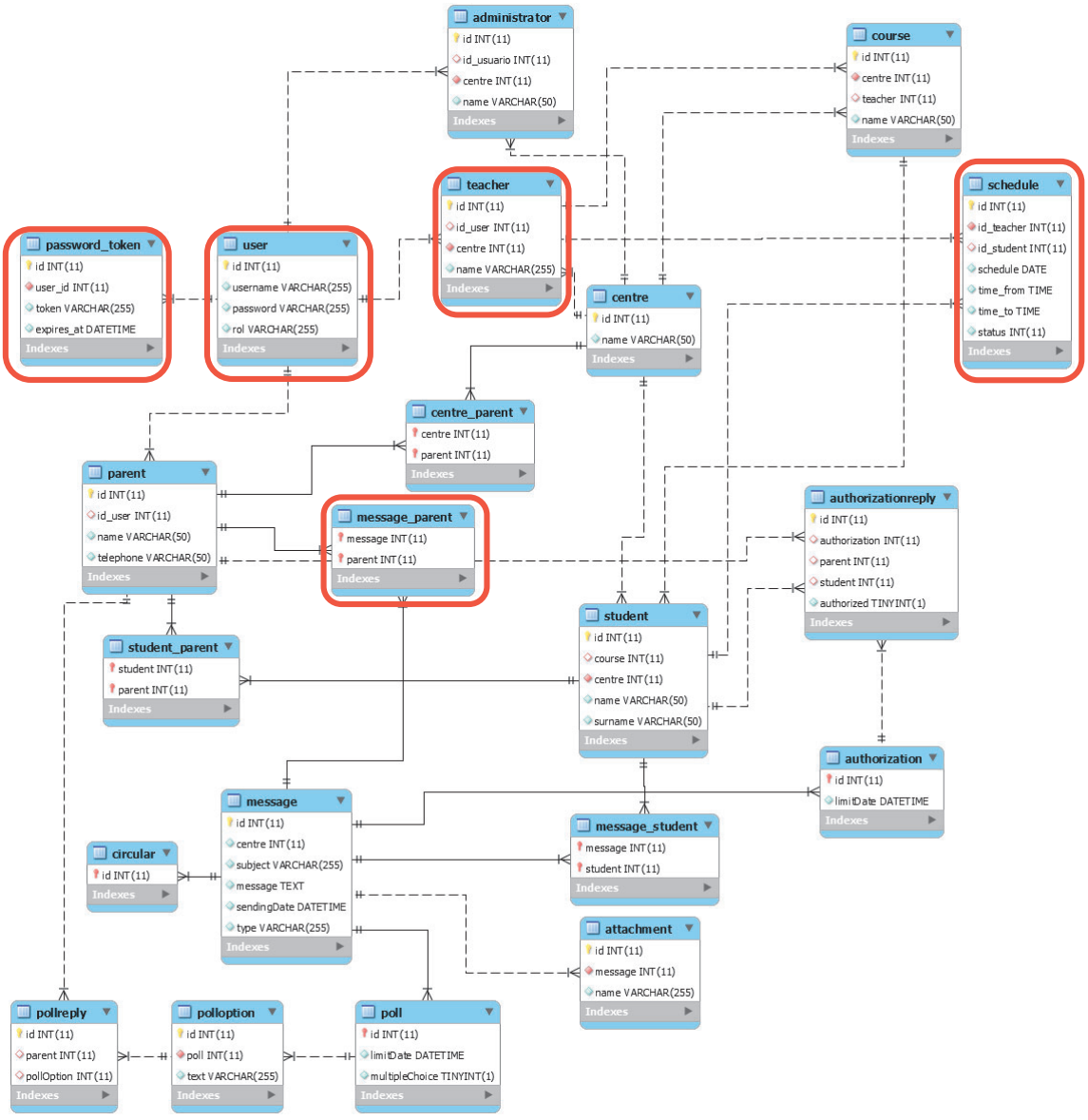


*Ilustración 14: Esquema base de datos*

Como se puede observar en la *Ilustración 14*, la actualización de la base de datos ha consistido en añadir cinco nuevas tablas, cuatro de ellas están relacionadas con entidades y una de ellas es una relación de entidades.

- La tabla definida como `user`, ha sido necesaria para implementar un registro e inicio de sesión común para todos los usuarios (administrador, profesor, padre/madre/tutor legal) que pueden realizar estas acciones en el sistema.
- La tabla definida como `password_token`, ha sido necesaria para implementar un sistema que permita al profesor restablecer su contraseña en caso de olvidarse de la misma.
- La tabla definida como `teacher`, ha sido necesaria para introducir en el sistema el rol del profesor.
- La tabla definida como `schedules`, ha sido necesaria para que el profesor pueda gestionar sus horarios de tutoría y sus citas.
- Por otra parte, se ha añadido una nueva tabla de relación de entidades denominada `message_parent`. Esta tabla ha sido añadida para tener un registro de que padres/madres/tutores legales han leído los mensajes enviados por el centro.

## 8.3 Almacenamiento de ficheros en el servidor.

En la primera fase del proyecto se decidió que los ficheros adjuntos a los mensajes (circulares, encuestas y autorizaciones) se guardarán en el sistema de ficheros por las siguientes razones:

- El nivel de conocimiento necesario para mantener una base de datos es proporcional al tamaño de la base de datos, ya que:
  - Dificulta las migraciones.
  - Dificulta las copias de seguridad.
- Guardar los ficheros en la base de datos complica el código encargado de guardar los ficheros en la base de datos, puesto que podemos guardarlos en distintos tipos de datos y tenemos que crear una capa de software distinta para cada tipo de datos.
- Es más complicado acceder a los ficheros guardados en una base de datos desde una aplicación web.
- No podemos aprovechar la potencia del almacenamiento en la nube si guardamos los ficheros en la base de datos.

Por tanto, en esta segunda fase del proyecto se ha decidido respetar esta decisión porque se considera que es una buena práctica no guardar los ficheros directamente en la base de datos.

## 8.4 Diseño arquitectónico

El *framework* utilizado para el desarrollo de esta aplicación utiliza un patrón de [27] arquitectura basado en componentes. En Angular 2, se pasó de utilizar una [28] arquitectura MVC (modelo-vista-controlador) a una arquitectura basada en componentes. Éstos son clases de *TypeScript* que incluyen tanto la lógica como la vista.

Además del uso de los propios componentes disponibles del *framework*, se han creado otros componentes, uno para cada vista, así como componentes individuales como lo son los componentes de las circulares, encuestas y autorizaciones.

Además, la lógica del componente trabaja con datos que vienen de diferentes proveedores existentes en la aplicación. Estos proveedores son los encargados de realizar las peticiones a la API REST para suministrar los datos a cada uno de los componentes que forma las vistas.

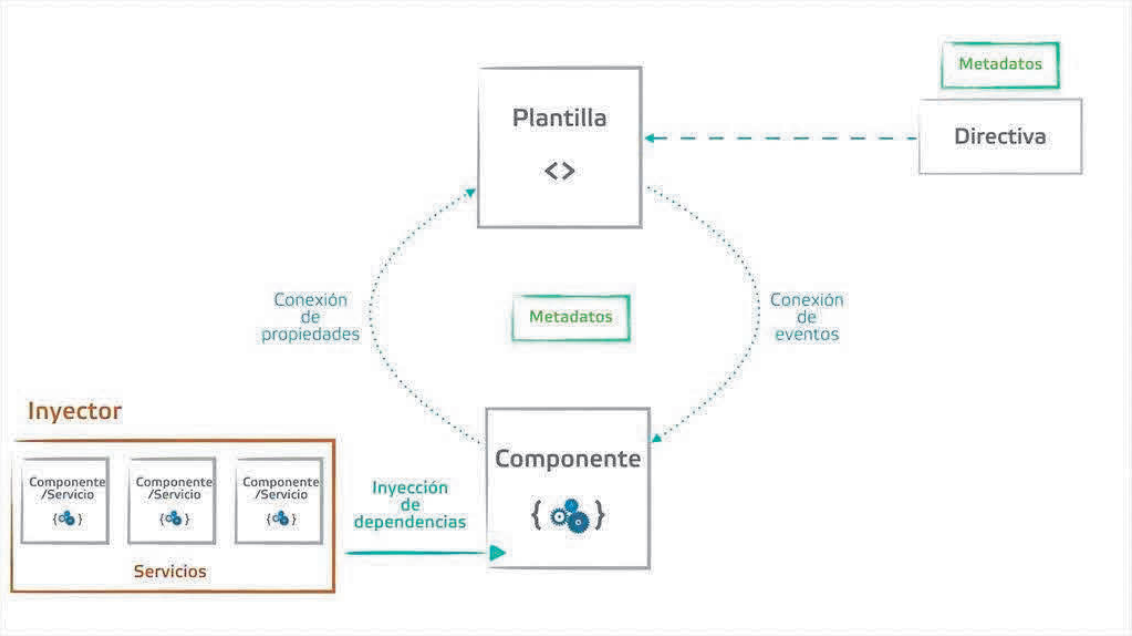


*Ilustración 15: Diagrama de arquitectura*

Como podemos ver en la *Ilustración 15* se identifican los ocho bloques de construcción principales de una aplicación *Angular*:

- Componentes: Un componente controla una sección específica de pantalla de la aplicación llamada *vista*.
- Plantillas: Se define la vista de un componente con una plantilla. Una plantilla contiene un fragmento de *HTML* que indica a Angular cómo renderizar el componente. Una plantilla se ve como *HTML* normal, con algunas diferencias.
- Metadatos: Los metadatos le indican a Angular cómo procesar una clase.
- Enlaces de datos: *Angular* soporta la vinculación de datos, un mecanismo para coordinar partes de una plantilla con partes de un componente. Agrega un marcado

de enlace de datos a la plantilla *HTML* para decirle a Angular cómo conectar ambos lados.

- Directivas: Las plantillas en *Angular* son *dinámicas*. Cuando Angular las transforma, convierte el *Document Objet Model* (*DOM*) de acuerdo con las instrucciones dadas por las directivas.
- Servicios: Son los proveedores de la aplicación. Los servicios abarcan una amplia categoría que incluye cualquier valor, función o característica que la aplicación necesite.
- Inyección de dependencias: La inyección de dependencias es una forma de suministrar una nueva instancia de una clase con las dependencias que ésta requiere. La mayoría de las dependencias son servicios. *Angular* utiliza la inyección de dependencias para proporcionar nuevos componentes con los servicios que necesitan.
- A todo esto, solo nos falta añadir los módulos. Las aplicaciones de *Angular* son modulares y Angular tiene su propio sistema de modularidad llamado módulos *Angular* o *NgModules*.

# 9. Desarrollo

## 9.1 IONIC 3

Primero explicaremos por qué se ha decidido la migración de *Ionic 1* a *Ionic 3* en el desarrollo.

El primer motivo importante lo encontramos en el cambio de *AngularJS* a *Angular 5*. ¿Cómo afecta este cambio?

- Mayor rendimiento y mayor optimización.

  La aproximación de *Angular* a flujo de información es unidireccional, en contraste con la vinculación de datos bidireccional que teníamos en *AngularJS*. Gracias a esto, la comprobación de cambios es más sencilla y además se consigue una mayor optimización.

  Nada que ver con los aparatosos cambios entre el modelo y la vista a través de ciclos de *AngularJS*, que se atragantaba cuando tenía que comprobar miles de enlaces.

- Aproximación más modular por componentes.

- Definición más simple de componentes y servicios mediante clases.

  Uno de los problemas con la curva de aprendizaje de *AngularJS* era la definición de los distintos elementos. Directivas, servicios, proveedor, etc. y cada cosa tenía su sintaxis y podía ser un poco abrumador para un recién llegado.

  *Angular 5* elimina en gran medida esos mecanismos ya que componentes y servicios se definen a través de clases.

- Inyección de dependencias simplificada con *TypeScript*.

  Gracias a *TypeScript* inyectar un servicio a un componente es tan simple como pasar su proveedor (la propia clase) como argumento al constructor del componente. Lo mismo con otros servicios.

  Se acabó lo de pasar un *array* de *strings* con las dependencias, muy dado a errores de escritura.

- Sintaxis de enlaces más clara.

Con *AngularJS*, nunca sabía lo que tenía que pasar en los atributos de las directivas sin echar un vistazo dentro: ¿Un literal? ¿un objeto? ¿una función?

*Angular* proporciona una sintaxis mucho más clara para los atributos de los componentes:

- `<item [something]="myVar">`: Corchetes para atributos de entrada
- `<item (onSomething)="myMethod()">`: Paréntesis para lanzar una función al suceder un evento
- `<item [(ngModel)]="myVar">`: Corchetes + paréntesis para un enlace bi-direccional
- `<item *ngFor="let item in items">`: No es propiamente un enlace al elemento, sino una directiva estructural (afecta a la estructura, en este caso repitiendo el elemento), y lo sabemos por qué se identifica con un *delante.

Con el *AngularJS* original no teníamos forma de saber la diferencia de un vistazo.

- Detección de errores en fase de "compilación" con *TypeScript*.

  El uso de variables tipadas permite detectar errores en fase de compilación al pasar de *TypeScript* a *Javascript* plano, ya que se puede saber si, por ejemplo, en un momento dado estamos esperando recibir un valor *booleano* y recibimos un *string.*

- Nuevo sistema de Navegación**.**

  El nuevo sistema de rutas mucho más adaptado al entorno móvil, donde hay una pila de vistas donde podemos añadir una encima (*push*) o sacarla de la pila (*pop*). Este sistema está más adaptado al comportamiento del usuario en un entorno de aplicación móvil.

### 9.1.1 Estructura del proyecto

*Ionic* 3 dispone de una estructura de directorios bien definida.

**`node_modules`**: La carpeta `node_modules` se genera automáticamente al instalar las dependencias *npm* con `npm install`. Este comando explora el archivo `package.json` para todos los paquetes que necesitan ser instalados. No necesitamos modificar nada en esta carpeta.

**`platforms`**: En esta carpeta se generará los proyectos nativos para cada plataforma que se hayan añadido previamente. Si hemos añadido la plataforma *iOS* y *Android* se

creará una carpeta llamada *iOS* y otra llamada *Android* y dentro tendrán los archivos y carpetas con la estructura de un proyecto nativo.

Estas carpetas se actualizan cada vez que se compila o se ejecuta el proyecto en un emulador o en un dispositivo. Salvo excepciones no tendremos que modificar nada en estas carpetas.

**`plugins`**: Contiene los *plugins* de *Cordova* que hayamos instalado. Se crean automáticamente en esta carpeta al instalar un *plugin* así que tampoco tendremos que modificar nada en esta carpeta a mano.

**`resources`:** Contiene el icono y la "*splash screen*" (pantalla de presentación) de la aplicación con la que después podremos crear automáticamente todas las imágenes en todos los tamaños necesarios para cada plataforma, lo que nos ahorrará mucho tiempo al no tener que generar a mano todos los tamaños de imagen necesarios del icono y la pantalla de presentación.

**`src`:** Esta es la carpeta más importante y donde realizaremos la mayor parte de nuestro trabajo. Aquí es donde están los archivos con el contenido de nuestra aplicación, donde definimos las pantallas, el estilo y el comportamiento que tendrá nuestra aplicación.

**`www`:** Esta carpeta se genera automáticamente y contiene la versión actual del código cada vez que efectuamos un cambio.

Veamos ahora varios archivos que se generan al crear un proyecto con *Ionic 3*.

**`config.xml`:** El archivo `config.xml` contiene parámetros que se utilizan cuando se construye un proyecto nativo a partir de un proyecto *Ionic*. Aquí deberemos indicar los permisos especiales que necesite la aplicación y otras configuraciones que puedan ser necesarias.

**`Ionic.config.json`:** Contiene información básica sobre la configuración de nuestro proyecto, se utiliza si es necesario subir la aplicación a la plataforma Ionic.io.

**`package.json`**: Contiene paquetes y dependencias de *nodeJS*.

**`tsconfig.json`** y **`tslint.json`:** Son archivos que contienen información necesaria a la hora de compilar *TypeScript*, no necesitamos editar estos archivos.

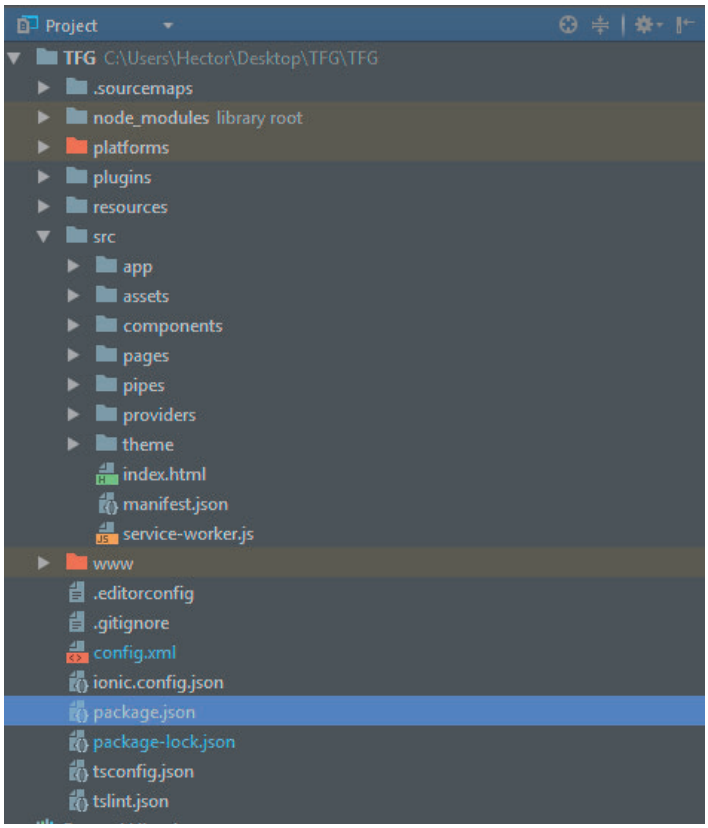


*Ilustración 16: Estructura de directorios*

Aunque pueda parecer complicado, en realidad la mayoría de los elementos los gestiona automáticamente *Ionic* y nosotros solo tenemos que preocuparnos de la carpeta `src` que es donde se va a situar nuestro código. Ocasionalmente puede que tengamos que editar algún archivo fuera del directorio `src` como el archivo `config.xml`.

Dado que el directorio `src` es el principal en el que debemos trabajar, procederemos a la explicación de cada uno de los subdirectorios.

- `Pages`: Contiene un directorio por cada página de la aplicación y contiene el componente que la forma.
- App: El archivo `app.component.ts`, define el componente principal sobre el cual se va construir la app. El archivo `app.module.ts`, es donde se tienen que importar todos los componentes que se vayan a usar en la app, tanto páginas completas, como partes de estas, así como servicios.
- `Assets`: Aquí es donde se suelen guardar las imágenes que se utilizan en la aplicación.
- `Components`: Aquí se definen componentes que serán usados en nuestras páginas.
- `Pipes`: En este directorio se crearán los filtros a usar en los componentes.

- `Providers`: Este directorio contiene todos los servicios que proveerán de datos a los componentes.
- `Theme`: En él se encuentra el fichero de variables globales de la aplicación, con la cual podremos cambiar fácilmente y de forma global los colores principales de la aplicación, por ejemplo.

### 9.2.2 Conexión con el backend

Para la conexión con el *backend*, con el cambio realizado sobre las peticiones a la API para mejorar la seguridad, es necesario enviar en las cabeceras de las peticiones un *token* de autenticación para poder acceder a los datos. Por lo que se ha creado un servicio llamado *`HttpUsingFormDataService`* el cual es el encargado de realizar los POST, GET, PUT y DELETE de todos los servicios de la aplicación.

El servicio *`HttpUsingFormDataService`*, hace uso del *`api_route`*, fichero en el cual se localiza la URL que apunta al *backend*, permitiendo que, durante el desarrollo, haya sido muy fácil cambiar del entorno de desarrollo al de producción. Al trabajar con un compañero, era necesario desacoplar las rutas la URL del *backend*, para poder alternar entre el trabajo realizado en local, y el código común colocado en producción en un servidor accesible desde internet.

## 9.2 Symphony

Para el desarrollo de la *API REST,* como ya se ha mencionado, se ha elegido *Symphony 3* como *framework* de desarrollo. En este apartado hemos trabajado sobre el proyecto del Trabajo Fin de Grado de D. Daniel Romero Calero titulado, 'ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UN BACKEND PARA LA COMUNICACIÓN ENTRE CENTROS EDUCATIVOS Y PADRES DE ALUMNOS II' tutorizado por el Dr. Alexis Quesada Arencibia, el cual ha realizado algunas de las API necesarias para la implementación del proyecto, y sobre este proyecto, se han implementado las *API* que faltaban para completar la funcionalidad.

### 9.2.1 Estructura del proyecto

Los directorios principales son los siguientes:

- `/app/`: ficheros de configuración de la aplicación, plantillas y traductores.
- `/bin/`: ficheros ejecutables (por ejemplo, la consola).
- `/src/`: el código PHP del proyecto (controladores, entidades, etc.).

- `/tests/`: tests.
- `/var/`: ficheros generados (cache, logs, etc.).
- `/vendor/`: dependencias.
- `/web/`: directorio raíz de la aplicación web. Aquí incluimos imágenes, archivos CSS y JavaScript, entre otros.

Para la realización de la API, solamente hemos tenido que dirigirnos al directorio `src` y dentro de este se encuentra el directorio **AppBundle**, en el que se encuentran los controladores, entidades, servicios y *normalizers* utilizados para el desarrollo de la API REST.

### 9.2.2 Enrutamiento

El enrutamiento en *Symfony* se puede implementar de distintas maneras, en este proyecto se ha decidido utilizar las anotaciones ya que ha resultado ser la manera más fácil y entendible de definir las rutas.

```
/**
 * @Route("/{id}/schedules", name="listarHorariosDeTutoriaProfesor")
 * @Method("GET")
 */
```

*Ilustración 17: Anotaciones para enrutamiento.*

Como se observa en la *Ilustración 17*, este sistema de creación de rutas es muy sencillo de entender y de aplicar ya que solamente debemos definir la ruta encima de la función del controlador y el método HTTP a utilizar.

Utilizamos peticiones HTTP donde cada método tiene un significado específico para manejar acciones CRUD (crear, leer, actualizar y borrar) utilizando métodos HTTP de la siguiente forma:

- GET: para devolver un recurso.
- POST: para crear un recurso.
- PUT: para actualizar un recurso.
- DELETE: para eliminar un recurso.

Para la creación de rutas hemos intentado que cada ruta tenga sentido desde una perspectiva del consumidor de la API. Por ejemplo, si lo que queremos es obtener un recurso, en este caso, un profesor, lo que debemos de llamar es el método GET, de dicho profesor de la siguiente forma: `GET -> '/teachers/{id}'`. Sin embargo, si lo que queremos es eliminarlo debemos usar el verbo DELETE: `DELETE->`

‘/teachers/{id}. Para crearlo o actualizarlo, debemos de seleccionar el método POST o PUT, según la acción a realizar, y en el cuerpo de la petición enviar los parámetros.

Para lidiar con las relaciones se ha seguido la siguiente estructura:

- GET: /parents/{id_parent}/students. Obtener los hijos asociado al padre ‘id_parent’.
- POST: /parents/{id_parent}/students/{id_student}. Asociar el estudiante ‘id_student’ al padre, ‘id_parent’.
- DELETE: /parents/{id_parent}/students/{id_student}. Desasociar el estudiante ‘id_student’ al padre, ‘id_parent’.

## 9.3 Seguridad

Dado que en la primera fase no se desarrolló ninguna medida de seguridad en la API, para esta fase se ha decidido implementar el uso de [29][30] *JSON Web Token (JWT).* Este es un estándar abierto basado en *JSON* para crear un *token* que sirva para enviar datos entre aplicaciones o servicios y garantizar que sean válidos y seguros.

Los ***JWT*** tienen una estructura definida y estándar basada en tres partes:

**header.payload.signature**

El ***header*** de un ***JWT*** tiene la siguiente forma:

```
{
"alg": "HS256",
"typ": "JWT"
}
```

La propiedad *alg* indica el algoritmo usado en la firma y la propiedad *typ* define el tipo de *token*, en nuestro caso ***JWT***.

El ***payload*** de un ***JWT*** es un *JSON* que puede tener cualquier propiedad, aunque hay una serie de nombres de propiedades definidos en el estándar.

```
{
"sub": 11,
  "id": 2,
  "name": "Padre 2",
  "username": "222222222",
  "rol": "progenitor",
```

```
  "iat": 1529536502,
  "exp": 1535584502
}
```

- `(sub)` - Es el id del usuario en la aplicación.
- `(id)` - Es el id según el rol del usuario.
- `(name, username, rol)` - Datos identificativos del usuario.
- `(exp)` - Una fecha que sirva para verificar si el JWT está vencido y obligar al usuario a volver a autenticarse.
- `(iat)` - Indica cuando fue creado el JWT.

Por último, la firma del ***JWT*** se genera usando los anteriores dos campos en base64 y una clave secreta (que solo se sepa en los servidores que creen o usen el ***JWT***) para usar un algoritmo de encriptación.

El proceso completo del *JWT* consta de estos pasos:

1. El usuario de una aplicación web/móvil/desktop hace un inicio de sesión con sus credenciales en el servidor donde esta publicada el *API*.
2. El usuario es validado en el servidor y se crea un nuevo *Token JWT* (usando nuestro "*secret-key*") para entregárselo al usuario.
3. El servidor retorna el *JWT* firmado que contiene los datos referentes al usuario y caducidad del *Token*.
4. El cliente/*browser* almacena el *JWT* para su uso y lo envía en cada petición mediante "`Authorization: Bearer`".
5. El servidor verifica la firma del *Token*, su caducidad y comprueba si el usuario tiene permisos para acceder al recurso leyendo los datos del *payload*.
6. El servidor responde al cliente la petición una vez que ha confirmado el Token y comprobado que los permisos del usuario son correctos.

# 10. Resultados, conclusiones y trabajo futuro

## 10.1 Resultados y conclusiones

Los objetivos de este proyecto se pueden dar por satisfechos, ya que se ha conseguido desarrollar un prototipo con todas las funcionalidades requeridas para facilitar las comunicaciones entre los centros y los padres. Se ha conseguido migrar todas las funcionalidades existentes en la anterior versión, y desarrollar mejoras en ellas, así como desarrollar el nuevo módulo de comunicaciones ente profesores y padres, con la gestión de tutorías.

En el transcurso del desarrollo de este Trabajo Fin de Título, hemos tenido que lidiar con un proyecto ya comenzado, teniendo que analizar el estado actual de dicho proyecto y de los competidores ya existentes. Gracias a los conocimientos obtenidos en la carrera, hemos podido analizar las alternativas existentes y estudiar detenidamente las mejorar que deberíamos implementar. Dichas mejoras han surgido no solo a nivel de funcionalidad, sino a nivel de tecnologías utilizadas, viendo que la decisión de migración del *framework* de desarrollo de la primera fase ha sido necesaria, ya que la versión actual con respecto a la que se trabajó en la fase anterior, dispone de infinidad de mejoras, y para continuar mejorando el proyecto era necesario dar este paso, aunque ello implicara rehacer mucho trabajo.

Durante el paso por la carrera, se trabajan todos los puntos necesarios para montar un gran proyecto, pero siempre de forma aislada. Al realizar este proyecto y tener que tocar todas estas fases en un único trabajo, nos damos cuenta de las capacidades obtenidas en la carrera. Esto es debido a que nunca había trabajado con tantas tecnologías en un solo proyecto, y tenido que hacer uso de todas las facetas de un desarrollo como este.

Al partir de un proyecto ya realizado también nos hemos dado cuenta de la influencia e importancia del avance de las tecnologías, ya que, en relativamente poco tiempo, el mundo del desarrollo de aplicaciones hibridas ha crecido a una velocidad descomunal, viendo como el *framework* de desarrollo utilizado en la versión anterior ya queda bastante obsoleto en el mercado actual.

Este trabajo también nos ha servido para darnos cuenta, que, aunque no tengamos experiencia en ciertas herramientas, como en este caso eran *symphony*, *angular*, *ionic*, etc. los conocimientos adquiridos en la carrera nos han permitido manejarnos con facilidad y poder afrontar un desarrollo de estas características en relativamente poco tiempo. En el grado hemos obtenido las bases para poder estructurar un proyecto y adaptarnos a las necesidades de cada caso.

## 10.2 Trabajo futuro

Aunque el estado actual del proyecto ya cubre muchas funcionalidades básicas y se puede poner en explotación, el proyecto es tan ambicioso, que aún quedan en el tintero

muchas mejoras por aplicar para poder ampliar la utilidad que tiene el sistema en el problema que se quiere resolver.

A continuación, listaremos una serie de mejoras que creemos que serían de mucha utilidad incorporarlas en un futuro en el sistema:

- Actualmente el sistema de acceso a los datos personales para la petición de portabilidad**,** que entró en vigor con la RGPD, se debe hacer mediante una petición al administrador del sistema. Se podría implementar una herramienta de solicitud directa de estos datos mediante la plataforma.
- Actualmente el sistema de comunicaciones padre-profesor, solo es posible con el tutor. Se podría diseñar un sistema de gestión de asignaturas por cursos, y profesores de cada asignatura, a los que después de tener una cita con el tutor, te pueda derivar con el profesor responsable de cada una de las asignaturas de los hijos.
- Sería conveniente implementar una herramienta de envió de mensajes móviles, para la autenticación de los padres.
- El desarrollo de notificaciones *push* para el aviso de proximidad de citas pendientes, o de nuevos mensajes disponibles.

# Bibliografía


[1] **FACTORYAPPS:** Aplicación para colegios. [en línea]. Disponible en: https://www.factoryapps.es/app-colegios/

[2] **Tokapp school:** Sistema de gestión para centros. [en línea]. Disponible en: https://www.tokappschool.com/welcome/centros

[3] **Tokapp school:** Aplicación para padres y alumnos. [en línea]. Disponible en: https://www.tokappschool.com/welcome/padres

[4] **Tokapp school:** Aplicación para profesores. [en línea]. Disponible en: https://www.tokappschool.com/welcome/profesores

[5] **miColegioApp:** Que es miColegioApp [en línea]. Disponible en: http://micolegioapp.com/wordpress/que-es-micolegioapp/

[6] **ClickEdu**: Plataforma de gestión para colegios [en línea]. Disponible en: https://clickartedu.com/inicio-plataforma-colegios.html

[7] **Remind:** Que es Remind [en línea]. Disponible en: https://www.remind.com/es-419/

[8] **ClassDojo:** ClassDojo y sus funcionalidades. [en línea]. Disponible en: https://www.classdojo.com/es-es/

[9] **Ionic1:** Documentación de *Ionic1* [en línea]. Disponible en: https://ionicframework.com/docs/v1/

[10] **Licencia de software:** Wikipedia licencia de software. [en línea]. Disponible en: https://es.wikipedia.org/wiki/Licencia_de_software

[11] **Licencia publica general GNU:** Wikipedia. GNU General Public Licence, [en línea]. Disponible en: https://es.wikipedia.org/wiki/GNU_General_Public_License

[12] **XAMPP:** Wikipedia. "XAMPP", [en línea]. Disponible en: https://es.wikipedia.org/wiki/XAMPP

[13] **Licencia comercial**: Wikipedia. "Software propietario", [en línea]. Disponible en: https://es.wikipedia.org/wiki/Software_propietario

[14] **Software Gratis**: Wikipedia. Software gratis, [en línea]. Disponible en: https://es.wikipedia.org/wiki/Software_gratis

[15] **Licencia MIT:** Wikipedia. "Licencia MIT", [en línea]. Disponible en: https://es.wikipedia.org/wiki/Licencia_MIT

[16] **Licencia PHP:** Wikipedia. “Licencia PHP”, [en línea]. Disponible en: https://es.wikipedia.org/wiki/Licencia_PHP

[17] **Licencia Apache:** Wikipedia. “Apache License”, [en línea]. Disponible en: https://es.wikipedia.org/wiki/Apache_License

[18] **Licencia JetBrains:** “TOOLBOX SUBSCRIPTION LICENSE AGREEMENT FOR EDUCATION”, [en línea]. Disponible en: https://www.jetbrains.com/student/license_educational.html

[19] **Reglamento General de Protección de Dato**s: REGLAMENTO (UE) 2016/679 DEL PARLAMENTO EUROPEO Y DEL CONSEJO de 27 de abril de 2016. [en línea]. Disponible en: https://www.boe.es/doue/2016/119/L00001-00088.pdf

[20] **Ley Orgánica de Protección de Dato**s: Wikipedia. “Ley Orgánica de Protección de Datos de Carácter Personal”, [en línea]. Disponible en: https://es.wikipedia.org/wiki/Ley_Org%C3%A1nica_de_Protecci%C3%B3n_de_Datos_de_Car%C3%A1cter_Personal_(Espa%C3%B1a)

[21] **Metodóloga de trabajo basada en prototipos:** Wikipedia. “Modelo de prototipos”, [en línea]. Disponible en: https://es.wikipedia.org/wiki/Modelo_de_prototipos

[22] ***Ionic* 3**: Documentación oficial de *Ionic* 3 [en línea]. Disponible en: https://ionicframework.com/docs/

[23] **API REST:** TSGroup. “La Arquitectura REST”, [en línea]. Disponible en: http://www.tsgroup.com.co/wps/portal/tsg/blog/detalle-blog/la-arquitectura-rest

[24] **API REST:** GitBooks. “Arquitectura de una API REST”, [en línea]. Disponible en: https://juanda.gitbooks.io/webapps/content/api/arquitectura-api-rest.html

[25] **Angular :** SG Buzz. “Angular: Mucho más que un framework”, [en línea]. Disponible en: https://sg.com.mx/revista/56/angular

[26] ***Symphony* 3:** Documentación de *Symphony 3* [en línea]. Disponible en: https://symfony.com/blog/category/documentation

[27] **Arquitectura Angular:** Angular. “Architectura overview”, [en línea]. Disponible en: https://angular.io/guide/architecture

[28] **Arquitectura MVC:** Wikipedia. “Modelo vista controlador”, [en línea]. Disponible en: https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador

[29] **JSON Web Token:** EnMiLocalFunciona. “Contruyendo una Web API REST segura con JSON Web Token”, [en línea]. Disponible en: http://enmilocalfunciona.io/construyendo-una-web-api-rest-segura-con-json-web-token-en-net-parte-i/

[30] **JSON Web Token:** JWT introducción. [en línea]. Disponible en: https://jwt.io/introduction/

# ANEXO I: Manual de usuario

En este anexo detallaremos las funcionalidades disponibles en la aplicación móvil, con un manual de usuario en el que se usarán las capturas de pantalla del dispositivo móvil para ir explicando en detalle cada funcionalidad contenida en la aplicación. Dado que la aplicación está destinada a dos roles separaremos el manual para cada uno de ellos.

## Apartado Padre/Madre/Tutor

### Inicio de sesión y registro

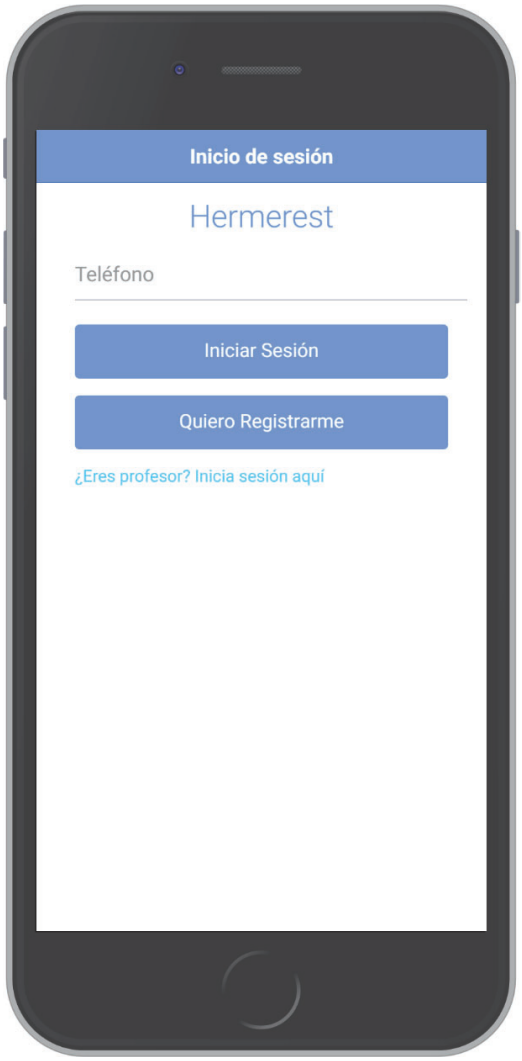


*Ilustración 18: Inicio de sesión.*

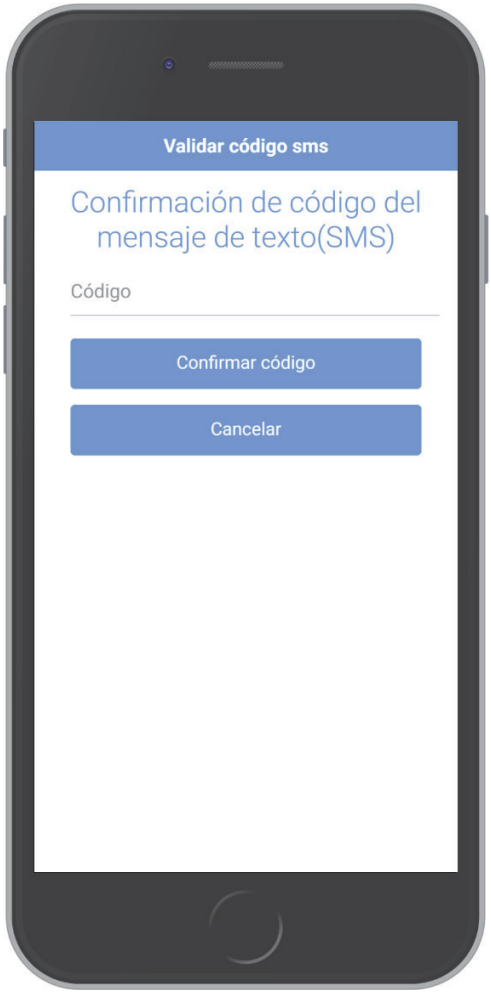


*Ilustración 19: Validación de código SMS.*

En la *Ilustración 18* observamos la pantalla con la que se encuentran los usuarios nada más acceder a nuestra aplicación. En ella podemos introducir nuestro teléfono móvil, si ya estamos registrados, y el sistema comprobará si ya existe para pasar a la vista de confirmación del código SMS, que podemos observar en la *Ilustración 19* (aunque esta funcionalidad aún no está desarrollada y el código por defecto seleccionado es del 123456). Este código pasaría a ser el código de seguridad para responder a las autorizaciones, aunque se puede modificar en la configuración del perfil del padre como veremos más adelante. Una vez confirmado ya habremos iniciado sesión en el sistema y podremos acceder a las demás funcionalidades del padre. Si aún no testamos dados de alta en el sistema tendremos que ir a la opción “Quiero Registrarme”.

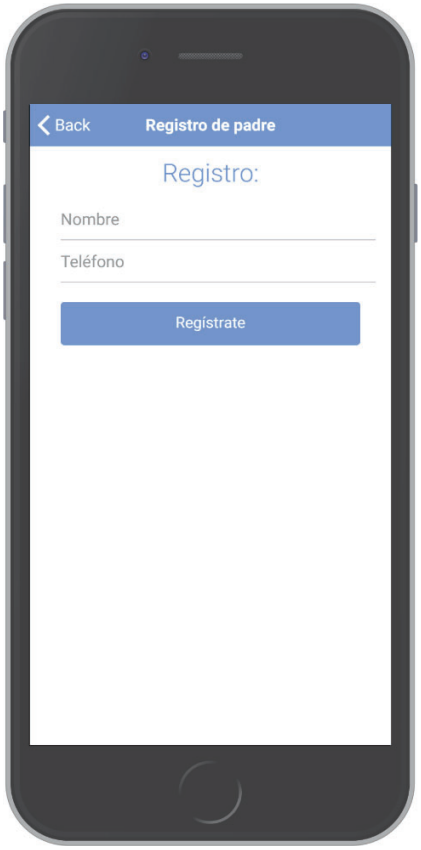


*Ilustración 20: Registro del padre.*

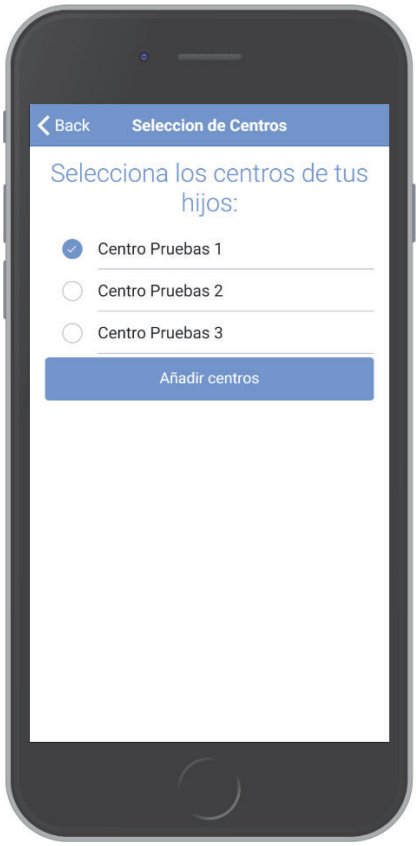


*Ilustración 21: Selección de centros.*

Para el registro de un padre en la plataforma debemos ingresar nuestro número de teléfono y el nombre. Una vez creado el usuario en el sistema se pasará a la selección de los centros (*Ilustración 21*) en los que está nuestro hijo, ya que, si no seleccionamos ninguno, no tendremos la opción de realizar ninguna acción en el sistema. Una vez seleccionados, saldrá un mensaje indicándonos que todo ha ido correctamente y que debemos ponernos en contacto con nuestros centros, para que nos asocien a nuestros hijos ya que son los administradores del centro los encargados de esta tarea.

## Ajustes


*Ilustración 22: Ajustes.*

Una vez dentro, en la parte superior derecha disponemos de los ajustes que puede realizar el padre. Saldrá el menú con las opciones (*Ilustración 22*), en las cuales podremos acceder al panel de mi perfil, al listado de hijos o volver a seleccionar los centros, ya que de un año a otro puede variar los centros en lo que se encuentran los hijos.


*Ilustración 23: Perfil del padre.*

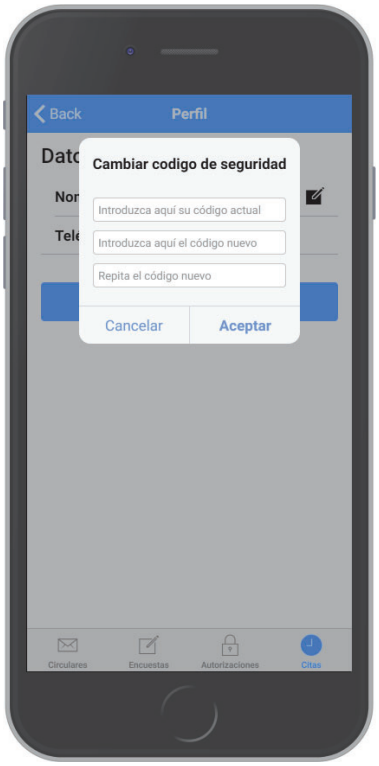


*Ilustración 24: Cambiar código de seguridad.*

En la sección del perfil podremos actualizar nuestro nombre o cambiar el código de seguridad necesario para confirmar las autorizaciones. Este sería el recibido por SMS al iniciar sesión (actualmente 123456 por defecto). Para cambiar de nombre saldría un diálogo similar, pero con el campo único del nombre.

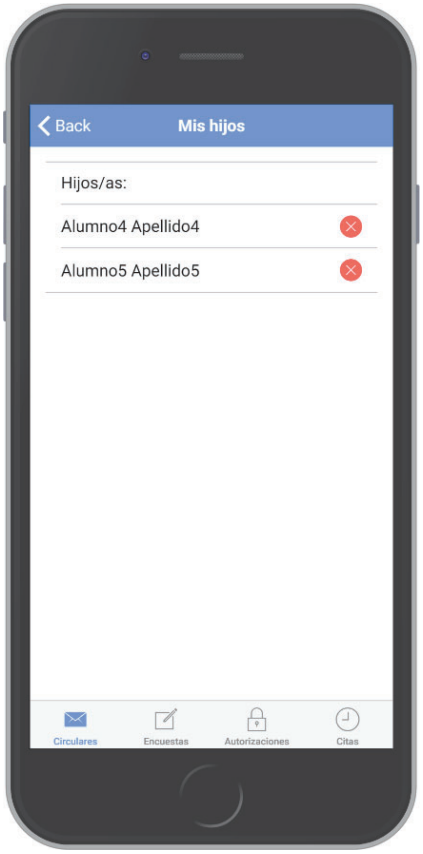


*Ilustración 25: Listado de hijos.*


*Ilustración 26: Diálogo desasociar hijo.*

En esta sección tendremos disponible el listado de hijos que ya tenemos asociados (*Ilustración 25*) y, por si hubiera algún error, podemos desvincular a uno de los alumnos asociados. Para ello saldrá un dialogo de confirmación indicando que esa acción es irreversible, por lo que se tendría que volver al centro si nos equivocamos eliminando uno de nuestros hijos (*Ilustración 26*).

## Circulares


*Ilustración 27: Listado circulares.*

Este apartado, a pesar de tener una interfaz muy sencilla, da toda la información necesaria al usuario. Esta vista se compone de un; buscador, para cuando tengamos una mayor cantidad de circulares no resulte tedioso buscar una en concreto y un listado con las circulares disponibles, así como las pestañas de navegación a las diferentes secciones de la aplicación.

Cada uno de los ítems del listado se compone por el título de la circular e información referente a ella, por ejemplo, si de un documento adjunto, representado por el icono derecho en forma de clip y, si ya hemos leído o no esta circular, para evitar confusiones

de si ya hemos accedido o no a una de las circulares, así como la fecha de envió de la circular, todo ello de forma muy visual.

Si accedemos a esta vista, entraremos en el detalle de la encuesta (*Ilustración 28*), en la cual, podremos entrar en detalles como, de qué centro proviene la circular, ya que podemos tener hijos en diferentes centros. Además, la fecha de envió de la circular, el contenido de dicha circular y si contiene un fichero adjunto, se muestra en azul, indicando que podemos descargarlo clicando sobre el link.

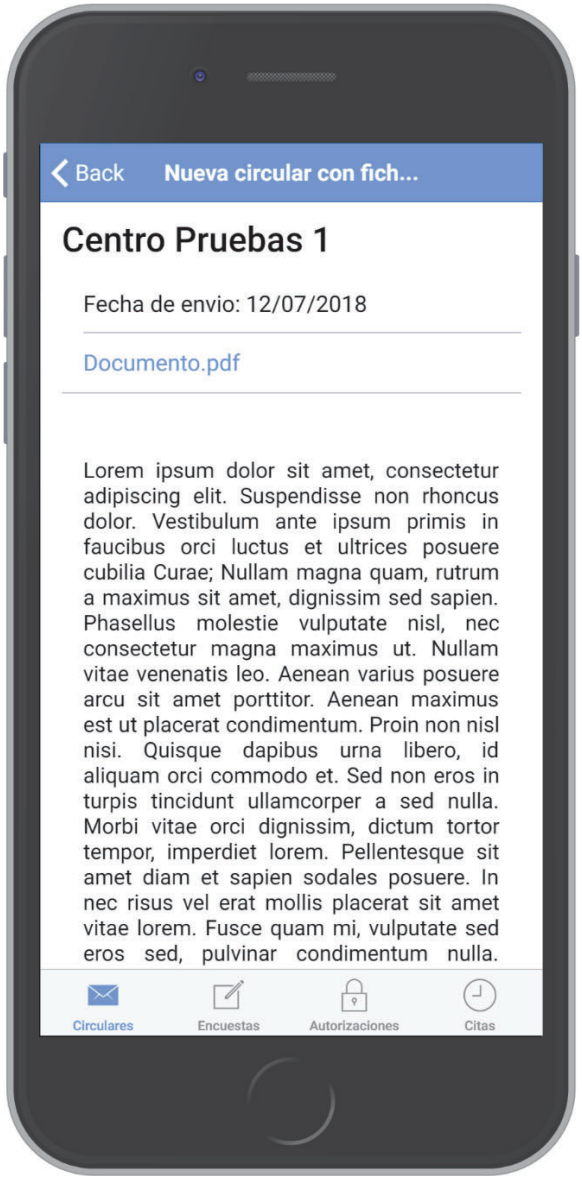


*Ilustración 28: Detalle circular.*

# Encuestas

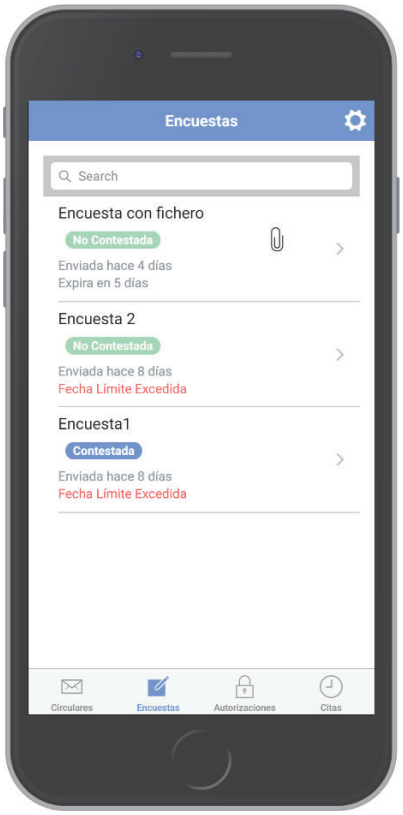


*Ilustración 29: Listado encuestas.*


*Ilustración 30: Detalle encuesta.*

El listado de encuestas es muy similar al de circulares, salvo que aquí se añade la fecha límite de la encuesta para su contestación o si ya se ha superado este tiempo, como podemos observar en la *Ilustración 29*.

El detalle de la encuesta es algo más elaborado que el de la circular como podemos ver en la *Ilustración 30*, ya que este añade las opciones disponibles y la fecha límite de respuesta. Una vez contestada la encuesta se eliminarán las opciones disponibles y el botón de enviar encuesta, mostrando un mensaje que indica que la respuesta ya ha sido contestada. Las opciones a marcar están definidas por el administrador del sistema y se permite la opción de múltiple selección o única selección según se haya definido el tipo de pregunta.

## Autorizaciones

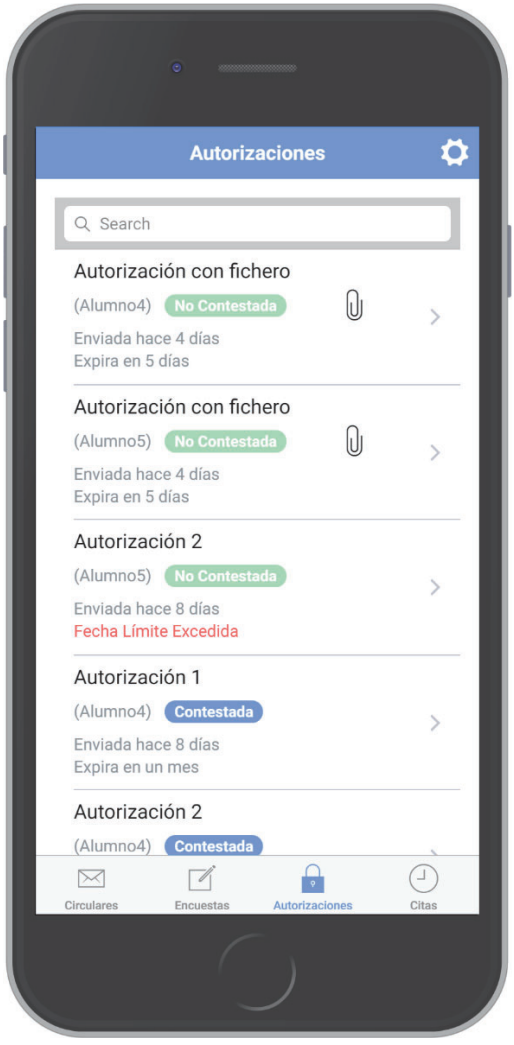


*Ilustración 31: Listado autorizaciones.*

En el listado de autorizaciones, la distribución es similar al de las encuestas, salvo que aquí se añade a que hijo va dirigida la autorización, evitando así tener que entrar en cada una de las autorizaciones para saber cuál de los hijos es el destinatario.


*Ilustración 32: Dialogo, confirmación acción.*

En cuanto al detalle de la autorización la acción de autorizar o denegarla, requieren de la inserción del código de seguridad por parte del padre, ya que es un tema muy delicado y en el que se requiere la certeza de que el padre es quien da la autorización. Para ello, se sigue el mismo modelo visto hasta ahora para estas acciones, presentar un modal de confirmación en el que debe introducir el código para validar la acción (*Ilustración 32*).

## Citas

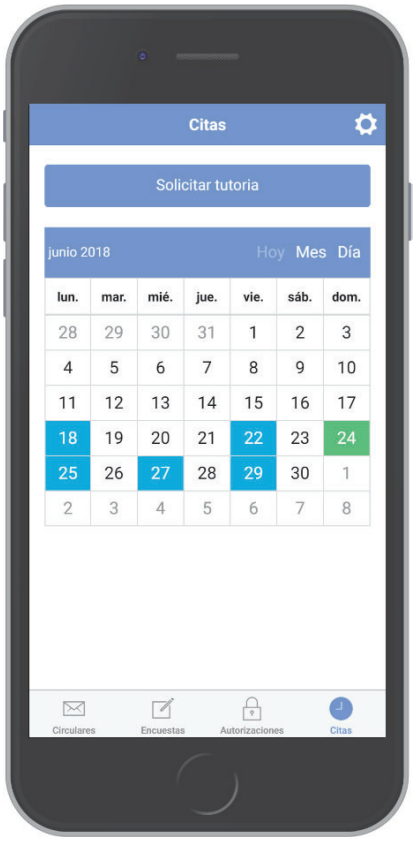


*Ilustración 33: Calendario de citas vista mensual.*


*Ilustración 34: Calendario de citas vista diaria.*

En la sección de citas, disponemos de un calendario, en el cual tendremos marcados las citas concertadas como eventos y podemos alternar a una vista más en detalle, con el modo 'día' (*Ilustración 35*), en el cual podremos ver la hora exacta de la cita, el profesor con el que esta concertada, y el hijo del cual se va a tratar en la cita. Si queremos pedir una nueva cita, disponemos de la opción 'Solicitar tutoría'.


*Ilustración 35: Solicitar cita.*

La opción antes mencionada nos llevara a una vista con el listado de profesores de cada hijo, y el calendario de horarios disponibles del que tengamos seleccionado. Este calendario dispone de las mismas opciones que el calendario anterior (vista mensual y diaria). Aquí, para solicitar una cita, debemos clicar en una de las franjas disponibles, y mediante un diálogo de confirmación realizar una petición de cita con el profesor.

# Apartado profesor

## Inicio de sesión y recuperación de contraseña

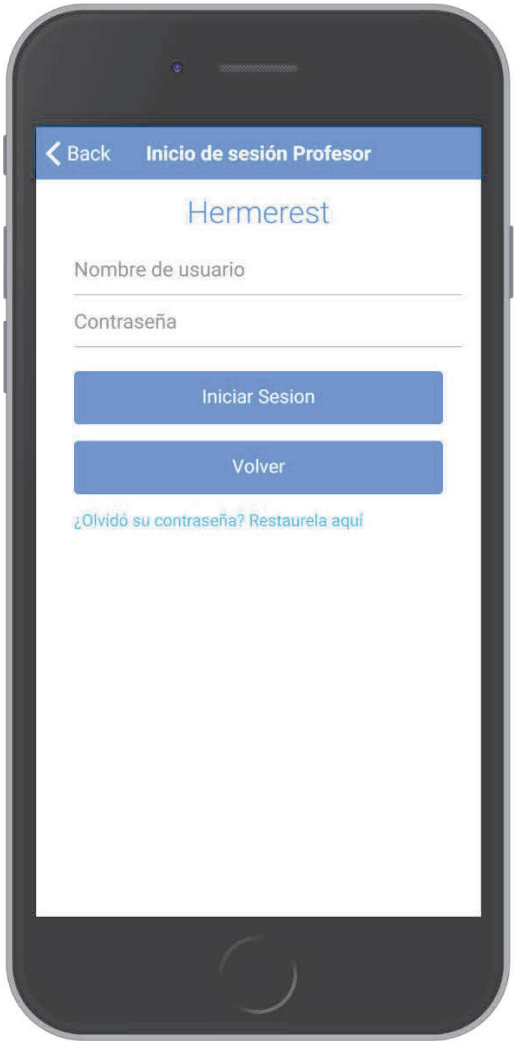


*Ilustración 36: Inicio de sesión del profesor.*

El inicio de sesión del profesor difiere del utilizado por los padres, puesto que el profesor, de lo que dispone es de una cuenta creada por el administrador del centro, el cual registra al profesor con un usuario (el correo institucional del profesor) y el docente asignará una contraseña mediante el correo que recibe al ser inscrito en el sistema. Una vez y dispone de cuenta, entrar en la plataforma es muy sencillo. Si vamos a la *Ilustración 18*, vemos como disponemos de un link el cual nos dirigirá a la vista que vemos en la *Ilustración 37*. Aquí solo debemos ingresar nuestro usuario y contraseña para entrar al sistema.


*Ilustración 37: Restauración de contraseña.*

Si no nos acordamos de nuestra contraseña y necesitamos restaurarla, es tan sencillo como ir al link situado debajo del inicio de sesión y realizar una petición de contraseña ingresando nuestro usuario (como mencionamos antes, el correo institucional del profesor). Siguiendo las indicaciones recibidas en el correo, podremos cambiar nuestra contraseña por una nueva.

## Ajustes

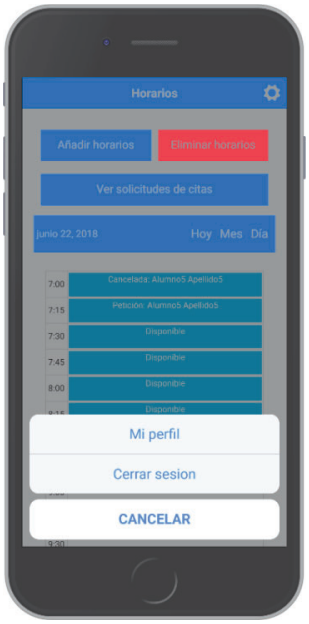


*Ilustración 38: Ajustes del profesor.*


*Ilustración 39: Perfil del profesor.*

Al igual que el padre, el profesor dispone de una sección de ajustes, aunque esta es mucho más limitada, ya que las opciones aquí disponibles son cerrar la sesión actual o acceder nuestro perfil (*Ilustración 40*). En nuestro perfil podemos cambiar nuestro nombre visible en la plataforma o hacer un cambio de contraseña. Para el cambio de contraseña se sigue el modelo establecido en toda la aplicación para estas acciones, se mostrará un diálogo, donde debe introducir los datos (en este caso, la antigua contraseña, y la nueva más su confirmación), y confirmar la acción.

## Calendario


*Ilustración 40: Calendario de horarios y citas vista mensual.*


*Ilustración 41: Calendario de horarios y citas vista diaria.*

El rol del profesor aparece por la necesidad de comunicar a padres y profesores, mediante un sistema de citas. Por ello, en nuestra aplicación el profesor dispone de un calendario con vista mensual y diaria (*Ilustraciones 41 y 42*), donde podrá ver cada una de las citas asignadas, pendientes, canceladas o los horarios aun disponibles. Para cada una de ellas podrá realizar diferentes acciones, en relación con el estado actual de la franja horaria. Es decir, si el estado actual es disponible, se accederá al detalle de este horario, permitiendo asignar una cita a uno de sus alumnos y estando disponible para sus padres o bien eliminar el horario (ver *Ilustración 42*). Sin embargo, si es una petición de cita se nos permitirá aceptarla o cancelarla, una cita cancelada se puede restaurar y una cita confirmada, se puede cancelar. Todas estas acciones se realizarán mediante un diálogo de confirmación de la acción.

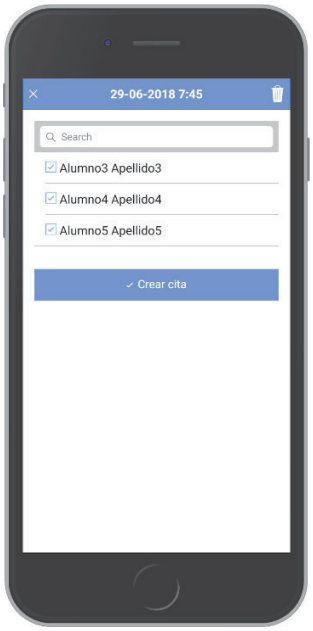


*Ilustración 42: Detalle horario disponible.*

## Gestión de horarios

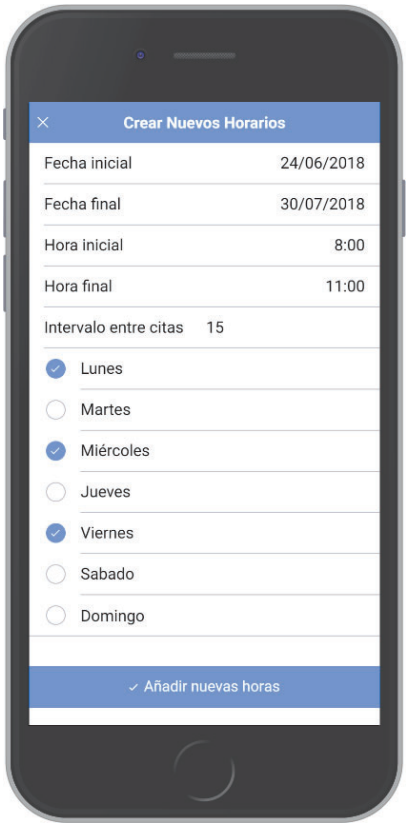


*Ilustración 43: Creación de nuevos horarios.*

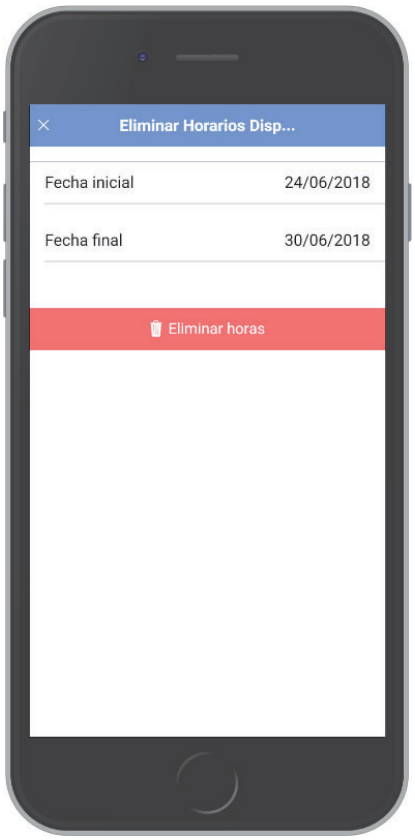


*Ilustración 44: Eliminado de horarios disponibles.*

Para la gestión de los horarios el profesor dispone de las opciones de crear nuevos horarios o eliminar horarios disponibles. En las ilustraciones 40 y 41 vemos desde donde se puede acceder a estas pantallas. La *Ilustración 43* es la vista donde el profesor va a poder configurar sus horarios de tutoría. Para ello, dispone de un formulario, para la generación de múltiples horarios, simplificando la tarea en un solo paso. Para definir los horarios de tutorías podrá seleccionar la fecha de inicio y de finalización del nuevo rango que se pretende añadir. Los siguientes campos son utilizados para determinar a qué hora se comenzará, la hora de finalización y el intervalo que tendrá cada una de las citas. Por último, debemos seleccionar que días de la semana tendremos disponible este horario. Así si queremos definir que todos los lunes y jueves, del mes de febrero, tendremos disponibles los horarios de 8:00 a 10:00 en franjas de 15 minutos, podemos realizarlo de una sola vez con dicho formulario.

La misma facilidad que se tiene para crear horarios disponibles, se debe tener para la eliminación. En la *Ilustración 44*, disponemos del formulario a realizar para eliminar los horarios disponibles entre dos fechas. Esta acción solo eliminará horarios disponibles, por lo que, si tenemos alguna cita ya concertada y debemos anularla, tendremos que ir al calendario y cancelarla manualmente.

## Solicitudes de citas

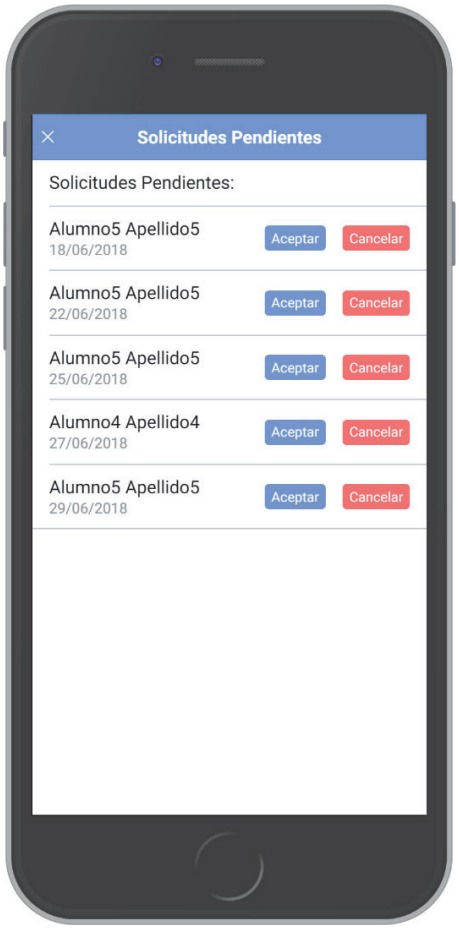


*Ilustración 45: Solicitudes de citas.*

Como la gestión de las peticiones de citas desde el calendario puede ser un poco tediosa, ya que tendríamos que ir día a día revisando si tenemos peticiones, se ha creado un apartado en el que gestionar todas las solicitudes sin necesidad de buscar en nuestro calendario. En la *Ilustración 45* vemos el listado de peticiones pendientes de aceptar o cancelar. Cada una de estas acciones mostrará un diálogo de confirmación.

# ANEXO II: Manual de instalación

Primero que nada, debemos instalar *NodeJS* ya que este cuenta con un manejador de paquetes llamado *NPM* el cual nos será necesario para instalar muchas de nuestras dependencias. Para instalar *NodeJS* debemos de dirigirnos la sitio oficial y descargar su instalador.

Para poder corroborar que la instalación fue satisfactoria y además para saber que versión instalamos de *NodeJS* y de *NPM* se pueden correr los comandos "`node -v`" y "`npm -v`".

```
C:\Users\Hector\Desktop\TFG\TFG> node -v

V8.9.2

C:\Users\Hector\Desktop\TFG\TFG> npm -v

5.5.1
```

A continuación, debemos instalar *Git* el cual es necesario para tener un control de versionado de nuestro proyecto y además es utilizado para descargar las librerías que necesita *Ionic* para ser ejecutado. Esto lo pueden descargar e instalar desde su sitio oficial y al igual que para *NodeJS* se puede corroborar si la instalación fue realizada correctamente ejecutando desde una terminal el comando "`git –versión`", el cual nos dará que versión hemos instalado en nuestro equipo.

```
C:\Users\Hector\Desktop\TFG\TFG> git --version

git version 2.11.1.windows.1
```

Por último, para completar el *setup* de *Ionic* instalaremos el *Ionic CLI* que es un complemento del *Cordova CLI* al que añade una serie de características adicionales. Para instalar *Apache Cordova* debemos hacerlo a través del manejador de paquetes NPM, lo que debemos hacer es abrir una terminar y correr el comando "`npm install -g cordova`"

Al instalarlo con la variable `-g` dispondremos de el de forma global y podremos acceder a él desde cualquier directorio. Una vez lo instalemos podemos verificar que todo haya salido correctamente corriendo el comando "`cordova -v`".

```
[C:\Users\Hector\Desktop\TFG\TFG> cordova -v

8.0.0
```

Luego de tener todo esto instalado procedemos a incorporar *Ionic Framework*, para realizar esto debemos abrir una terminal e ingresar como administrador para luego correr el comando "`npm install -g ionic`". Una vez esté instalado corroboramos como en casos anteriores que versión tenemos, esto lo podemos ver ejecutando el comando "`ionic -v`".

```
C:\Users\Hector\Desktop\TFG\TFG> ionic -v

3.20.0
```

Una vez instaladas las herramientas, podemos proceder a descargar el código del repositorio de *GitHub* "https://github.com/hectorfgf/TFG", y con un terminal, dirigirnos a la ruta donde tengamos nuestro proyecto y ejecutar la orden "`npm install`", la cual traerá todas las dependencias del proyecto, registradas en el fichero `package.json` de nuestro proyecto.

Dado que el proyecto se ha realizado conjuntamente con un compañero encargado de realizar el *backend*, la guía de instalación del *backend* se encuentra en el TFG titulado "ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UN BACKEND PARA LA COMUNICACIÓN ENTRE CENTROS EDUCATIVOS Y PADRES DE ALUMNOS II"

Una vez tengamos las dependencias instaladas, el *backend* montado, solo nos quedará un paso para tener el proyecto en funcionamiento. Para ello debemos dirigirnos a la carpeta "`src/providers`" de nuestro proyecto, y modificar la ruta de la *API* en el fichero "`api-route.ts`".

Ahora si tenemos el proyecto listo para ser usado. Para ello debemos ejecutar la orden "`ionic serve`" para poder usar la aplicación desde nuestros navegadores accediendo a la dirección "http://localhost:8100".