ULPGC
Universidad de
Las Palmas de
Gran Canaria

Escuela de
Ingeniería Informática

Grado en Ingeniería Informática

Trabajo Fin de Grado

Aplicación móvil para la visualización del consumo eléctrico de los clientes

ANDRÉS CABRERA DÍAZ

Tutor

Javier Sánchez Pérez

Las Palmas de Gran Canaria
Junio de 2023

Agradezco el apoyo recibido durante estos años
A mi madre,
A mis tíos,
A mis amigos cercanos,
A los compañeros con los que he trabajado durante la carrera y me han apoyado,
Y a mi tutor por darme la oportunidad de acompañarme durante este proyecto.

Tabla de contenido

Índice de Figuras

Índice de Tablas

Resumen

Con este trabajo final de carrera se pretende ofrecer de cara al usuario medio una plataforma para dispositivos móviles que, de manera intuitiva y amigable, permita visualizar los suministros a su nombre (independientemente de la distribuidora que los gestione) y permitir para cada uno de ellos, acceder al detalle del contrato y poder visualizar los consumos en diferentes periodos.

Esto será posible gracias a las posibilidades de acceso a la información que proporcionan el despliegue de contadores inteligentes llevado a cabo por las empresas de distribución eléctrica, las cuales ofrecen un amplio abanico de posibilidades a los clientes. **Datadis** es la plataforma de datos de consumo que proporciona la **Asociación de Empresas Eléctricas (ASEME)**, la cual dispone de una **API** privada de la cual se obtendrá la información de los suministros.

Finalmente, en cuanto a las tecnologías que se han utilizado a lo largo de la implementación de este trabajo han sido **Dart**, un lenguaje de programación gratuito y abierto a la comunidad, empleando para ello del "framework" de **Flutter** y en cuanto al lado del servidor se ha utilizado el gestor de base de datos de **SQLite**.

Abstract

With this final degree project, the aim is to provide an intuitive and user-friendly platform for mobile devices to the average user. This platform will allow users to view their utility supplies (regardless of the distribution company managing them) and access detailed contract information, as well as visualize consumption data for different periods.

This will be made possible thanks to the information access capabilities provided by the deployment of smart meters carried out by electric distribution companies, which offer a wide range of possibilities to customers. **Datadis** is the consumption data platform provided by the "**Asociación de Empresas Eléctricas (ASEME)"**, which has a private *API* from which the supply information will be obtained.

Finally, regarding the technologies used throughout the implementation of this project, **Dart**, a free and open-source programming language, has been employed. The **Flutter** framework has been used for mobile app development, while the **SQLite** database management system has been used on the server side.

Capítulo 1 Introducción

La viabilidad de este proyecto se debe a la disponibilidad de información proporcionada por los contadores inteligentes desplegados por las empresas de distribución eléctrica. Para obtener los datos, se ha hecho uso de **Datadis**[1], la plataforma de datos de consumo proporcionada por la **Asociación de Empresas Eléctricas (ASEME)**, que cuenta con una **API**[2] privada para obtener la información de los suministros.

Es importante destacar que la plataforma desarrollada en este trabajo no solo brinda a los usuarios una manera fácil y accesible de visualizar sus suministros eléctricos y consumos independientemente de la distribuidora que los gestione, sino que también les permite monitorear y controlar su consumo de energía eléctrica de una manera eficiente. Al poder acceder a la información de sus suministros y consumos eléctricos, los usuarios pueden tomar decisiones más informadas en cuanto al uso de la energía eléctrica, lo que puede resultar en un ahorro significativo en sus facturas de electricidad.

Permitiendo de dicha manera proporcionar de una herramienta accesible para el usuario medio, que permita visualizar los detalles de sus contratos y consumos en los diferentes períodos.

Por otro lado, la reducción del consumo de energía eléctrica no solo beneficia a los usuarios en términos económicos, sino también en términos ambientales. Al utilizar menos energía eléctrica, se reduce la huella de carbono y se contribuye a la sostenibilidad del medio ambiente. Por lo tanto, la plataforma desarrollada en este proyecto final de carrera tiene un impacto positivo tanto en el ámbito económico como en el ambiental.

En resumen, la plataforma desarrollada en este trabajo ofrece a los usuarios una solución sencilla y fácil de usar para visualizar sus suministros eléctricos y consumos, y al mismo tiempo les permite monitorear y controlar su consumo de energía eléctrica, lo que puede resultar en un ahorro significativo en sus facturas de electricidad y una reducción de la huella de carbono.

Para concebir dicha idea, se ha tratado de siempre tener en cuenta que el uso e información a mostrar debe ser clara, concisa y fácil de comprender para el usuario, además de adaptable. Es por ello, que se han aplicado los principales principios de diseño “Responsive”.

En cuanto a las tecnologías utilizadas dichas han sido **Flutter** debido entre otros factores a que permite un desarrollo simultáneo rápido para aplicaciones móviles multiplataforma (*iOS* y *Android*) haciendo uso únicamente de un código fuente, consiguiendo con ello reducir el tiempo y costes del desarrollo que implicaría desarrollarlo de forma nativa para ambas plataformas. Además, permite un diseño personalizable ya que al estar conformado por elementos denominados “*Widgets*”, los cuales pueden ser adaptables para crear diseños atractivos y funcionales. Por último, dicho lenguaje pese a ser relativamente nuevo, cuenta con una comunidad activa de desarrolladores que aportan paquetes y herramientas para mejorar la experiencia de desarrollo y cuenta por su parte con el respaldo de **Google**, compañía detrás de **Flutter**.

[1] Datadis: www.datadis.es
[2] API: Interfaz de programación de aplicaciones

En resumen, **Flutter** es una excelente opción para el desarrollo de este proyecto debido a su rapidez, flexibilidad, alto rendimiento y una comunidad activa y comprometida con el desarrollo de la herramienta.

Otra de las tecnologías utilizada para el entorno del servidor ha sido **SQLite**, la cual es una opción excelente para el almacenamiento de datos en aplicaciones móviles debido entre muchas cosas a su tamaño reducido y muy eficiente para aplicaciones móviles con limitaciones de almacenamiento y potencia. Además de ser compatible con sentencia **SQL** lo que permite realizar operaciones de *CRUD*[3] usando dicho lenguaje. Y como característica más importante hay que destacar que es una herramienta de almacenamiento de datos transaccional lo que permite garantizar las operaciones de escritura y lectura de forma segura y evitando la corrupción de los datos.

En un primer momento se había decidido implementar el almacenamiento de datos haciendo uso de **Firebase** gestor que incluye **Google** de manera sencilla se puede incorporar a los proyectos, pero dicha opción tiene una serie de inconvenientes tales como alto coste de conexión al ser una plataforma en la nube, empeorando los costes de operaciones de lectura y escritura. Dicha opción sería ideal si la aplicación necesitará una alta escalabilidad y un almacenamiento en la nube seguro y confiable, pero también puede tener un costo adicional y puede ser menos adecuado para aplicaciones más pequeñas o que no requieren una gran cantidad de servicios adicionales, como es el caso de dicho proyecto.

En resumen, **SQLite** es una excelente opción para el almacenamiento de datos en aplicaciones móviles debido a su ligereza, eficiencia, seguridad y compatibilidad con múltiples plataformas. Además, al ser compatible con **SQL** y su fácil integración con otros lenguajes de programación, los desarrolladores pueden utilizar **SQLite** para realizar operaciones de *CRUD* en sus aplicaciones móviles de forma eficiente y segura.

1.1 Motivación

Con este proyecto se persigue tener varias aportaciones significativas en distintos campos entre los cuales hay que destacar:

- **Investigación**: el proyecto podría aportar nuevas perspectivas y conocimientos en el campo de la monitorización y análisis de los consumos eléctricos, especialmente en lo que se refiere a la visualización y accesibilidad de los datos. También podría servir como base para futuras investigaciones en el campo de la eficiencia energética y el consumo responsable.
- **Social**: la aplicación podría ayudar a fomentar una mayor conciencia sobre el consumo eléctrico y la importancia de ahorrar energía, lo que podría tener un impacto positivo en el medio ambiente y en la sostenibilidad a largo plazo. Además, al ofrecer una herramienta más accesible para el usuario medio, la aplicación podría ayudar a empoderar a los consumidores (dado el potencial de la aplicación para mejorar la experiencia, sin tener especial relación con ideologías o regulaciones de políticas energéticas) y mejorar su experiencia con los suministros eléctricos.
- **Educativo**: la aplicación podría utilizarse como una herramienta educativa para enseñar a los usuarios sobre los diferentes aspectos de su contrato eléctrico, los diferentes tipos de tarifas y los consumos en diferentes periodos. Esto podría ayudar a mejorar la comprensión de los usuarios sobre los suministros eléctricos y promover hábitos de consumo responsables.

[3] CRUD: acrónimo de operaciones para realizar operaciones de creación, lectura, actualización y borrado.

- **Económico**: la aplicación podría contribuir a una mayor transparencia en el mercado eléctrico y a la competencia entre las diferentes distribuidoras. Al permitir a los usuarios comparar sus consumos y tarifas con diferentes proveedores, la aplicación podría ayudar a los usuarios a tomar decisiones más informadas y a elegir el proveedor que mejor se adapte a sus necesidades. Además, al fomentar el ahorro de energía, la aplicación podría ayudar a los usuarios a reducir sus facturas eléctricas y mejorar su situación financiera.

Por último, se podría aportar al concepto de "Business Intelligence" (BI)[4]. En el contexto de este proyecto, la aplicación móvil recopila y presenta información relevante sobre el consumo eléctrico de los usuarios. Esta información puede ayudar a los usuarios a tomar decisiones informadas sobre cómo administrar su consumo de energía y reducir sus costos.

El uso de BI implica la recolección, integración y análisis de datos empresariales con el objetivo de proporcionar información valiosa para la toma de decisiones. En este proyecto, la aplicación móvil se basa en la integración de datos de los contadores inteligentes y su presentación de forma intuitiva para los usuarios.

La aplicación también permite analizar el consumo de energía en diferentes períodos de tiempo, lo que permite a los usuarios identificar patrones de consumo y hacer ajustes para reducir el consumo de energía en momentos de mayor demanda. Estos análisis y ajustes pueden llevar a una reducción del costo de energía para los usuarios.

1.2 Objetivos

Entre los objetivos que perseguía lograr dicha aplicación se encuentran los siguientes:

- Facilitar a los usuarios finales el acceso a la información detallada sobre los consumos de sus suministros eléctricos, lo que les permitiría tomar medidas para reducir su consumo y, por tanto, un posible ahorro en su factura eléctrica.
- Proporcionar una plataforma de visualización de datos atractiva e intuitiva global que permita a los usuarios entender mejor su consumo eléctrico y su contrato con la distribuidora eléctrica.
- Ayudar a las empresas de distribución eléctrica a mejorar la gestión de la energía, proporcionándoles información detallada sobre el consumo eléctrico de sus usuarios.
- Contribuir a la mejora de la eficiencia energética y la sostenibilidad, fomentando la reducción del consumo eléctrico y, por tanto, la disminución de emisiones de CO_2.
- Servir como herramienta de educación y concienciación sobre la importancia de la gestión del consumo eléctrico y su impacto en el medio ambiente.

[4] "*Business Intelligence"* o inteligencia empresarial, hace referencia al conjunto de estrategias, a través del análisis de los datos. [30]

1.3 Organización del documento

Los apartados en los que se dividirán a partir de ahora dicho documento serán los siguientes:

- En el primer capítulo, se introduce de forma resumida la idea, finalidad y objetivos de este proyecto.
- En el segundo capítulo, se realizará una introducción al campo que remarca este proyecto, así como definiciones generales que se deben saber, para facilitar su lectura. Y se definen conceptos técnicos del mercado de la electricidad.
- En el capítulo tercero, se realiza un estudio del arte relacionado con plataformas que comparten alguna similitud con dicho proyecto y explicar brevemente en que consisten las mismas.
- En el capítulo cuarto se describe la metodología y la planificación empleada a lo largo del proyecto, además se analizará si se ha cumplido la planificación inicial planteada.
- En el capítulo quinto, se detalla la planificación del proyecto exponiendo las herramientas, recursos, lenguajes y base de datos utilizadas.
- En el capítulo sexto se desarrollará en torno al desarrollo propio de la aplicación, o sea, en otras palabras, el análisis, diseño, la implementación y despliegue de la misma.
- En el capítulo séptimo, se expondrán las conclusiones finales alcanzadas tras el desarrollo del proyecto y futuros funcionalidades o trabajos relacionados con el proyecto.
- Se incluye en el Anexo I las competencias que se han implementado en este proyecto.
- En el Anexo II el manual de usuario, donde se explica los requisitos y un recorrido a través de la aplicación.
- En el Anexo III se detallan el resto de las historias de usuario que no se han expuesto durante el transcurso del documento.
- En el último, Anexo IV se incluyen extractos clave del código implementado. HistoriasHistorias de usuario
- Finalmente se incluirá la Bibliografía en la cual se citan las fuentes consultadas a lo largo del desarrollo y han servido para el aprendizaje o implementación del trabajo.

Capítulo 2 Análisis del mercado eléctrico en España

El mercado de la electricidad en España es un sector dinámico y diverso que ha experimentado importantes cambios en los últimos años. Este mercado se compone de diversos agentes, entre los que se incluyen generadoras, distribuidoras, comercializadoras y consumidores. Todos estos agentes vienen regulados a través de la **Comisión Nacional de los Mercados y la Competencia (CNMC)** bajo la supervisión en estos momentos del "Ministerio para la Transición Ecológica y el Reto Demográfico", dicho departamento varía frecuentemente dependiendo del gobierno en control del mismo, comúnmente conocido como "Ministerio de Industria, Energía y Turismo".

España cuenta con una amplia diversidad de fuentes de generación eléctrica, habiendo sido históricamente dependiente de la energía nuclear y los combustibles fósiles como el gas natural y el carbón. Sin embargo, en los últimos años ha habido un notable incremento en la generación renovable, especialmente a través de la energía eólica y la solar fotovoltaica.

El mercado mayorista de electricidad, conocido popularmente como "pool eléctrico", es un mercado organizado donde los generadores ofertan su producción y los comercializadores adquieren energía para satisfacer la demanda de los consumidores. El precio de la electricidad se determina mediante un sistema de subastas y fluctúa cada hora en función de la oferta y la demanda.

Los consumidores tienen la posibilidad de elegir entre diferentes comercializadores de electricidad, que ofrecen una variedad de tarifas y contratos. Pueden optar por tarifas reguladas o contratar en el mercado libre, donde existe una mayor competencia y se ofrecen tarifas más personalizadas.

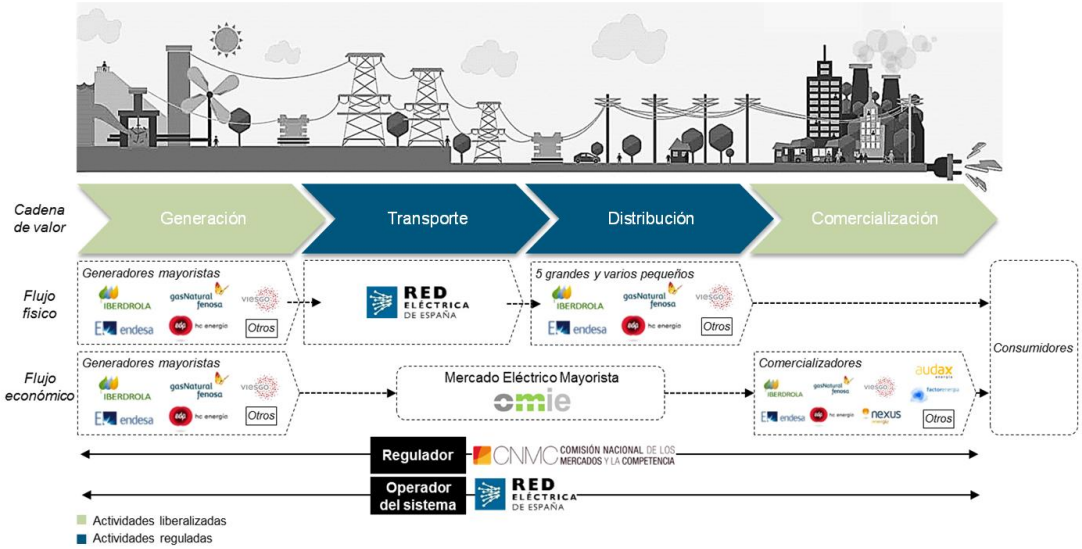


Figura 1 - Mercado Eléctrico Español. Fuente: Energía y Sociedad [1]

A continuación, se muestra un esquema del funcionamiento del mercado eléctrico español, en el que aparecen todos sus agentes intermediarios en el ciclo de vida desde su generación hasta su comercialización y llegada al consumidor.

- **Generación**: empresas que producen y venden la energía eléctrica.
- **Transporte**: la empresa propietaria de los elementos de la red, es decir, cableado, transformadores y dispositivos de energía reactiva. Dichas tareas son gestionadas por el monopolio de **Red Eléctrica de España** (REE).
- **Distribución**: empresas que operan las redes de distribución, en su caso suelen ser las mismas que operan la generación de la misma, además de otras más pequeñas.
- **Comercialización**: empresas que compran la energía en el mercado mayorista y la vende a los consumidores que no se les permite, participar directamente en el mercado mayorista.

El mercado a su vez, se divide en dos modalidades, el mercado libre y el mercado regulado conocido también como "pool eléctrico". En el primero de ellos, los agentes tienen la libertad de negociar directamente sus contratos de compra y venta de energía, lo que fomenta la competencia entre las distintas empresas. Por otro lado, en el mercado regulado, los precios de la electricidad se determinan mediante un sistema de subastas, denominado como "subasta marginalista". En este mercado, los consumidores acogidos al suministro regulado tienen tarifas establecidas por el Gobierno o la autoridad competente. Ambos mercados coexisten, gracias a ellos ofreciendo diferentes opciones a los consumidores y generando dinamismo en el sector eléctrico en España.

Red Eléctrica de España, es la empresa estatal que desempeña el papel fundamental en el esquema. Como operador del sistema único, su principal responsabilidad es gestionar y garantizar el suministro eléctrico seguro, eficiente y sostenible a lo largo de la red.

Además de su función operativa, REE desempeña un papel estratégico en la planificación y desarrollo del sistema eléctrico. Trabaja en estrecha colaboración con otros actores del sector, como generadores, distribuidores, comercializadores y organismos reguladores, para garantizar la eficiencia y fiabilidad del sistema.

Además, su relación con la regulación del mercado regulado, dicho organismo es el encargado de la gestión del mismo y el encargado de establecer los precios y organizar las subastas. Por ende, está involucrada de igual manera con el desarrollo y gestión del **Precio Voluntario al Pequeño Consumidor** (PVPC). El PVPC es un mecanismo de tarificación regulado por el Gobierno español que establece el precio de la electricidad para los consumidores acogidos al suministro regulado.

Para ello REE está encargado de proporcionar la información necesaria para su cálculo. Dicho valor varía según los datos de oferta y demanda de electricidad, así como la generación procedente de diferentes fuentes. En base a esa información, se establecen los precios de la electricidad para cada momento del día divididos a su vez en tres distintos tramos [2], los cuales se explicarán en la siguiente sección más en profundidad, conformando así los distintos periodos durante el transcurso del día.

El PVPC permite a los consumidores acogidos al suministro regulado beneficiarse de los precios mayoristas reales de la electricidad, ya que se basa en los costes marginales de producción. Esto significa que los consumidores pueden ajustar su consumo a las horas de menor demanda y, por lo tanto, a los precios más bajos, lo que fomenta la eficiencia energética y la reducción de costos.

De forma resumida, **Red Eléctrica de España** desempeña un papel crucial como operador del sistema eléctrico, asegurando el suministro y la calidad de la electricidad en España. Su trabajo abarca la operación y control de la red de transporte, la gestión del mercado mayorista regulado y la promoción de la transición hacia un sistema energético más sostenible.

2.1 Conceptos Técnicos

En este epígrafe se expondrán una serie de términos o tecnicismos de cara a dar un contexto al tema a tratar en este proyecto.

- **Mercado regulado**: mercado en el que los precios y las condiciones de los contratos de la electricidad está regulado por el Gobierno u organismo competente en este caso REE, donde los consumidores acogidos a este modelo tienen tarifas establecidas y no tienen libertad de elegir su proveedor de electricidad.

- **Mercado libre**: mercado en el que los precios y las condiciones de contratación de la electricidad son negociados directamente entre los consumidores y los comercializadores de electricidad. En este mercado, los consumidores tienen la libertad de elegir entre diferentes proveedores de electricidad y pueden negociar contratos personalizados según sus necesidades.

- **PVPC o Precio Voluntario al Pequeño Consumidor**: mecanismo de tarificación regulado por el Gobierno español para los consumidores acogidos al suministro regulado. El PVPC establece los precios de la electricidad en función de los costos marginales de producción y refleja las fluctuaciones en el mercado mayorista. Permite a los consumidores beneficiarse de los precios reales de la electricidad en cada hora del día.

- **Comercializadoras**: son empresas que se dedican a la venta de electricidad a los consumidores. Las comercializadoras pueden operar tanto en el mercado regulado como en el mercado libre, ofreciendo diferentes tarifas y contratos de suministro eléctrico. Son responsables de la facturación y atención al cliente.

- **Distribuidoras**: son empresas encargadas de la distribución de electricidad a los puntos de consumo. Mantienen y operan las redes de distribución, que incluyen líneas eléctricas y subestaciones, para entregar la electricidad a los hogares, negocios e industrias. Las distribuidoras son responsables de garantizar la calidad y continuidad del suministro eléctrico.

- **Contador eléctrico**: comúnmente denominado contador inteligente, es un dispositivo utilizado para medir el consumo de electricidad en un punto de suministro. El contador eléctrico registra la cantidad de electricidad consumida y se utiliza para calcular el importe de la factura eléctrica. Los contadores eléctricos pueden ser analógicos o digitales y transmiten los datos de consumo a las distribuidoras para su facturación.

En la actualidad el modelo más usado es el de contador eléctrico digital, el cual dispone de una serie de ventajas:

- **“Más rapidez en caso de avería:** Cualquier problema con tu suministro será más fácil de identificar y pasarás menos tiempo con la luz cortada.” (“Contadores inteligentes | Endesa”)

- **“Ajustar la potencia es más fácil:** Si quieres subir o bajar la potencia contratada ya no hace falta que un técnico vaya a tu casa.” (“Contadores inteligentes | Endesa”)
- **Adiós a las lecturas estimadas:** Todas son reales.
- **“Nada de manipulaciones:** Los contadores inteligentes están constantemente monitorizados y no pueden ser trucados.” (“Contadores inteligentes | Endesa”)

Figura 2 - Representación contador digital. Fuente: Endesa

En cuanto al funcionamiento de un contador eléctrico digital, este registra la cantidad de electricidad que fluye a través de un punto de suministro, mediante la detección de corriente eléctrica que pasa a través del circuito interno del contador a través de una unidad de medida estándar, generalmente en kilovatios-hora (kWh). Por lo general tienen asociado una pantalla, que muestra la lectura actual del consumo.

En cuanto a la transmisión de estos, suelen estar equipados con funciones de comunicación bidireccional que permite transmitir los datos del consumo a la compañía que lo opera. Lo que evita tener que realizar una lectura manual y por ende facilita su facturación.

En la siguiente ilustración, se puede apreciar un ejemplo de un contador eléctrico digital operada por Endesa, en el cual el contador cuenta con un panel digital en el que se muestra en el lado izquierdo el periodo de integración horaria, es decir si es hora punta (1) o valle (2) acompañado de la lectura de energía activa en kilovatios (kW).

Figura 3 - Modelo de contador digital de Endesa. Fuente: Endesa

En el contexto del suministro eléctrico además encontramos los términos de “hora punta” y “hora valle”, dichos términos refieren a diferentes periodos de tiempo durante un día en los cuales las demandas de electricidad pueden variar.

- **Hora punta**: se refiere al período de tiempo en el que hay una alta demanda de electricidad. Generalmente, esto ocurre durante las horas del día en las que la mayoría de las personas están activas, como por la mañana temprano y por la tarde-noche. Durante las horas punta, la demanda de electricidad puede ser más alta debido al uso generalizado de electrodomésticos, sistemas de calefacción o refrigeración, iluminación y actividades industriales. Los precios de la electricidad en las horas punta tienden a ser más altos debido a la mayor demanda y la necesidad de gestionar la carga en la red eléctrica.
- **Hora valle**: es el período de tiempo en el que la demanda de electricidad es más baja. Por lo general, ocurre durante las horas de la madrugada o durante el día cuando la

mayoría de las personas están ausentes o no están usando una gran cantidad de electricidad. Durante las horas valle, la demanda disminuye, lo que puede resultar en precios más bajos de la electricidad. Algunas compañías eléctricas ofrecen tarifas especiales para los consumidores que realizan un consumo importante durante las horas valle, incentivando el uso de electricidad en esos momentos de menor demanda.

A modo de conclusión el mercado eléctrico español ofrece a los consumidores la posibilidad de elegir entre el mercado regulado y el mercado libre, brindando diferentes opciones y tarifas para satisfacer sus necesidades. Además, el mecanismo del Precio Voluntario al Pequeño Consumidor (PVPC) permite a los consumidores acogidos al suministro regulado aprovechar los precios basados en la oferta y demanda de electricidad en cada hora del día. **Red Eléctrica de España** (REE) desempeña un papel fundamental como operador del sistema eléctrico y gestor del mercado mayorista. Las comercializadoras y distribuidoras cumplen funciones clave en la venta y distribución de electricidad. Además, comprender los horarios de hora punta y hora valle permite a los consumidores adaptar su consumo para aprovechar tarifas más bajas durante las horas valle, contribuyendo a un uso más eficiente y económico de la electricidad. En conjunto, los elementos expuestos en este apartado conforman el mercado eléctrico español, fomentando la competencia, la sostenibilidad y la participación activa de los consumidores en la gestión de su consumo de electricidad.

Capítulo 3 Estado del Arte

La obtención de datos a través de los contadores eléctricos o vatihorímetros se ha convertido en una pieza clave en la transición hacia un sistema energético eficiente, permitiendo un cambio significativo en la medición de los consumos de los puntos eléctricos. A pesar de ello, no se ha explotado completamente su uso para ofrecer beneficios al usuario final, quien podría ser el más beneficiado por esta información.

Generalmente, estos datos sólo son accesibles a través de la página web de la distribuidora eléctrica que abastece el punto o consultando los precios de la energía en distintos portales oficiales, como el de **Red Eléctrica de España**. Sin embargo, este proceso puede resultar tedioso para el usuario y requiere de un conocimiento previo de la distribuidora correspondiente.

Este proyecto tiene como objetivo fusionar ambos conceptos para ofrecer una herramienta al usuario final. En los siguientes capítulos, se analizarán algunos proyectos y alternativas existentes para abordar esta necesidad.

3.1 Plataformas análogas

Consulta a través de la web o aplicación de la propia distribuidora

La opción más común y tradicional que utilizan los usuarios para acceder a su información de suministro eléctrico es a través de la página web o aplicación móvil proporcionada por su distribuidora de energía. Sin embargo, esta opción presenta una desventaja, ya que los usuarios deben conocer a qué distribuidora pertenecen para poder acceder a su información.

Esto puede ser un inconveniente, especialmente para aquellos usuarios que se mudan a una nueva ubicación y no están familiarizados con las distribuidoras locales. Además, este método de acceso a la información puede no ser el más intuitivo y fácil de usar para algunos usuarios, lo que podría dificultar la comprensión de los detalles de su contrato y consumos de energía. Por lo tanto, se requiere una herramienta más accesible y amigable para el usuario medio, que permita visualizar la información de suministro eléctrico de manera intuitiva e independiente de la distribuidora que gestione los suministros.

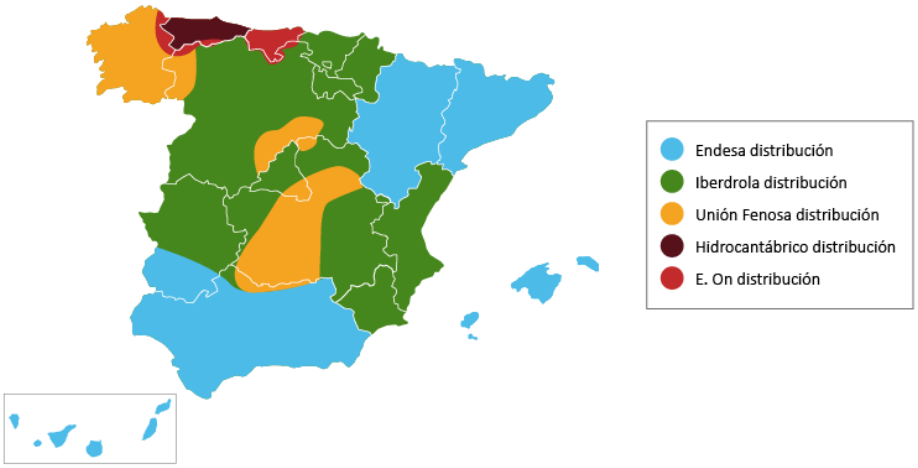


Figura 4 - Distribuidoras eléctricas España. Fuente: Autosolar [2]

Endesa Clientes es una de las plataformas que proporciona una de estas distribuidoras, Endesa a través de (e-distribución) a sus clientes. A través de esta plataforma, los clientes pueden de manera conveniente y eficiente administrar sus suministros y acceder a diferentes servicios.

Al registrarse en la plataforma, los clientes de Endesa pueden realizar diversas acciones, como consultar y descargar sus facturas eléctricas, realizar pagos en línea, cambiar su modalidad de contratación, gestionar sus datos personales y de contacto, y comunicarse directamente con el servicio de atención al cliente.

Además, la plataforma de **Endesa Clientes** brinda a los usuarios información detallada sobre su consumo de electricidad, incluyendo gráficos y datos históricos que les permiten analizar patrones y realizar un seguimiento de su consumo en el tiempo. Esto ayuda a los clientes a tener un mayor control sobre su consumo energético y a tomar decisiones informadas para optimizar su eficiencia y reducir costos.

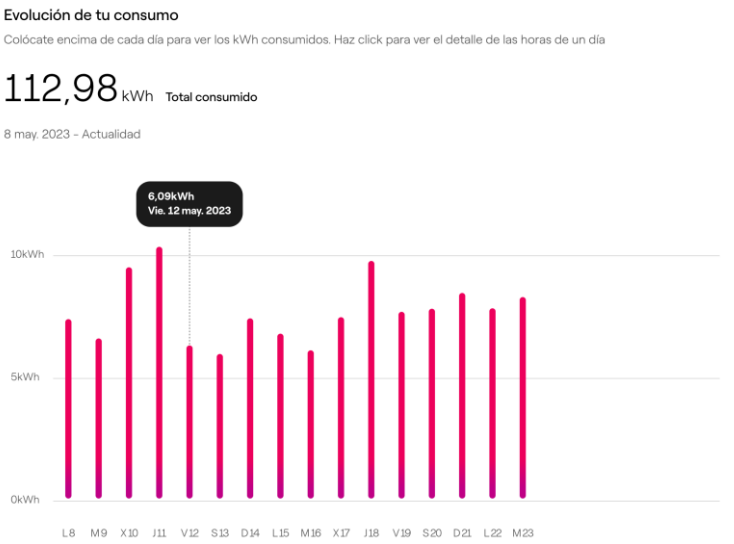


Figura 5 - Visualización de consumo a través de Endesa Clientes

3.2 RedOS

RedOS es la aplicación oficial de **Red Eléctrica de España** (REE), en la cual se ofrece tanto para consumidores y profesionales, generando un cuadro de mando diferente, donde poder consultar entre otros, distintos datos tales como:

- Precios minoristas
- Demanda eléctrica
- Generación eléctrica
- Emisiones de CO_2

La principal limitación es esa misma, únicamente permite visualizar información mínima y generalista que, al usuario medio, no aporta gran información para el beneficio y uso diario.

A pesar de esto, la interfaz de usuario es fácil de usar e intuitiva, por lo que se ha utilizado como modelo para el desarrollo del proyecto, tomando como base sus componentes visuales.

Además de **RedOS**, existen diversas aplicaciones que se pueden utilizar para monitorizar el consumo de energía en el hogar. Las características de estas aplicaciones incluyen la capacidad

de realizar un seguimiento de los gastos de energía, recibir alertas sobre el uso excesivo de energía, y comparar el uso de energía con otros hogares de la misma área.

No obstante, uno de los mayores desafíos que presentan estas aplicaciones es la falta de una interfaz intuitiva que resulte fácil de usar para la mayoría de los usuarios.

Asimismo, la necesidad de ingresar manualmente los datos de consumo en algunos casos puede resultar tediosa para los usuarios, lo que puede llevar a que muchos opten por no utilizar este tipo de aplicaciones.

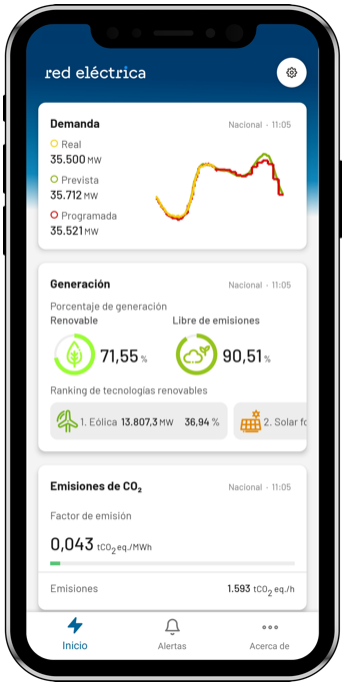


Figura 6 - Aplicación RedOs. Fuente: Red Eléctrica de España

3.3 DATADIS

Datadis [3] es una empresa que ofrece soluciones y servicios relacionados con la gestión y el análisis de datos. A través de su plataforma, la cual está diseñada para ayudar a las organizaciones a aprovechar al máximo sus datos y convertirlo en información valiosa para la toma de decisiones, permite a los usuarios consumidores acceder a información detallada y actualizada sobre su consumo de energía eléctrica y su coste. Los usuarios pueden consultar y descargar información sobre su consumo diario, semanal, mensual y anual, lo que les permite conocer y analizar sus hábitos de consumo y tomar decisiones informadas para reducir su consumo y ahorrar en sus facturas de energía. Además, la plataforma también permite a los usuarios comparar su consumo con el de otros usuarios similares y recibir recomendaciones personalizadas para reducir su consumo de energía. También pueden recibir alertas en caso de que se detecte un consumo anómalo o excesivo, lo que les permite identificar posibles problemas en sus instalaciones eléctricas y actuar rápidamente para evitar costes adicionales.

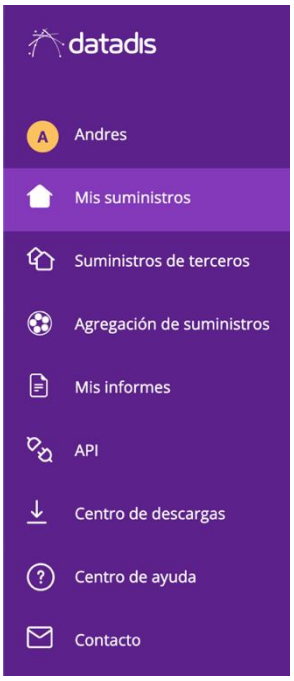
Algunas de las características claves y servicios que ofrece son:

- **Centralización de datos**: permitiendo a los usuarios integrar y consolidar datos de diferentes fuentes en un único lugar, lo que facilita la gestión y el acceso a la información.
- **Análisis y visualizado de datos**: ofrece además capacidad para realizar análisis de datos avanzados, permitiendo visualizar los datos a consultar de manera clara y comprensible utilizando gráficos y tablas.
- **Generación de cuadros de mando**: permite generación de informes detallados en diferentes formatos de descarga.
- **API privado**: dispone además de una completa API privada para poder realizar consultas a información de consumo almacenada en las bases de datos de las distintas distribuidoras.

En resumen, la plataforma de **Datadis** permite a los usuarios consumidores tener un mayor control y conocimiento sobre su consumo de energía eléctrica y su coste, lo que les ayuda a tomar decisiones más eficientes y ahorrar en sus facturas de energía.

En cuanto a su utilización es muy sencilla, únicamente debemos de disponer de algún suministro a nuestro nombre o estar autorizado en alguno, y registrarnos en el portal de plataforma. Una vez realizado este paso ya podremos consultar toda la información relativa a los suministros.

Una vez nos conectamos a través de la plataforma encontramos un menú vertical en el lado izquierdo de la pantalla con diferentes accesos directos a las funcionalidades de la plataforma, las cuales se dividen de la siguiente manera:



- **Perfil**: en este primer apartado podremos visualizar la información relativa a la cuenta de usuario en la plataforma. Con información como el nombre, apellidos, correo electrónico, contraseña, y configuraciones de la plataforma como el cambio de idioma.
- **Centro de descargas**: acceso a informes que hayamos solicitado o generado y podremos descargar una vez estén disponibles.
- **Centro de ayuda**: un acceso donde podremos consultar preguntas frecuentes, sobre la utilización de la plataforma.
- **Contacto**: apartado donde podremos formular consultar al equipo de desarrollo, cabe destacar que el mismo funciona muy bien, ya que se ha hecho uso del mismo.

Figura 7 – Menú de Datadis

- **Mis suministros**: en este apartado se muestran los suministros a nuestro nombre, dichos datos son obtenidos a través de los datos de las distintas distribuidoras. A través de aquí podremos consultar el contrato, acceder a la página de la distribuidora, visualizar el consumo o descargar datos del mismo. También disponemos de un apartado para visualizar aquellos suministros a los que hemos autorizado a otros usuarios.

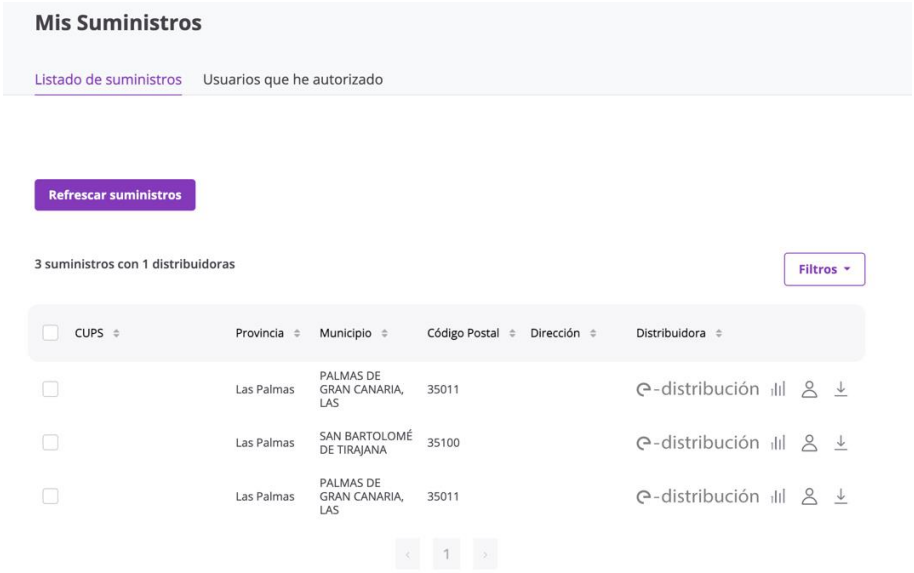


Figura 8 - Sección de "Mis suministros" en Datadis

- **Suministros de terceros**: en el caso de que nos hayan autorizado a visualizar algún suministro, los mismos aparecen en este apartado, de forma similar a como lo hacen en el apartado anterior.

- **Agregación de suministros**: en caso de que la plataforma no detecte automáticamente un consumo, en este apartado podremos añadir el mismo proporcionando los datos a través de un formulario.
- **Mis informes**: Podremos a base de un formulario generar informes de datos almacenados en las bases de datos, como para, por ejemplo, comparar consumos en distintos niveles.
- **API**: punto por el cual se accede a la información almacenada en las bases de datos, dicho apartado debido a su importancia será tratado en un apartado separado.

A continuación, se muestran dos ilustraciones (Tabla 1 - Cuadros de mandos en Datadis) en las cuales podemos observar cómo genera la plataforma los elementos del cuadro de mandos para la visualización de los consumos.

Figura 9 - Cuadro de mandos informe consumos en Datadis

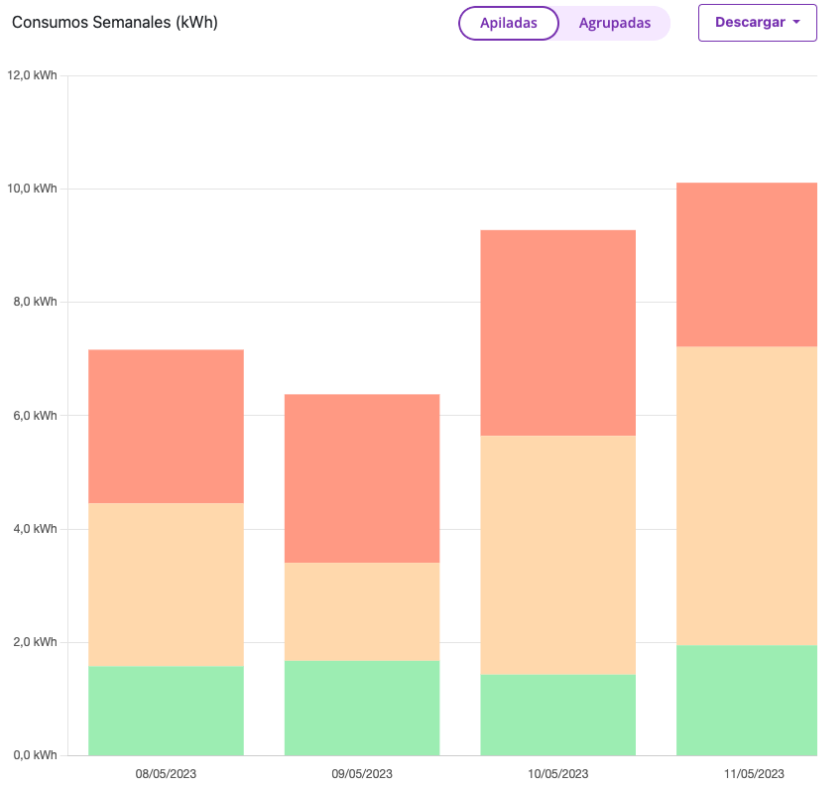


Figura 10 – Cuadro de mandos gráficos de consumos en Datadis

Tabla 1 - Cuadros de mandos en Datadis

Capítulo 4 Planificación del Proyecto

4.1 Metodología

Siguiendo los conceptos aprendidos durante este período formativo, se ha decidido implementar la metodología *SCRUM* aplicando además la metodología de *Kanban* conjuntamente. A continuación, se expondrá en qué consisten ambos y cómo se han aplicado durante la realización de este proyecto.

SCRUM es una metodología ágil la cual consiste en un conjunto de buenas prácticas desarrolladas con el fin de obtener mejores resultados de planificación durante la ejecución de un proyecto. Entre sus principales características se hallan las entregas parciales también denominadas Sprint, con esto se consigue aportar valor de manera incremental al cliente del producto, ya que podrá ir visualizando las funcionalidades previamente definidas y verificarlas. Por otro lado, también aporta que sea idóneo para entornos cambiantes o variables donde los requisitos finales no están completamente definidos, como es el caso de este proyecto, en el cual mediante se iba desarrollando iban apareciendo nuevos requisitos funcionales o no funcionales.

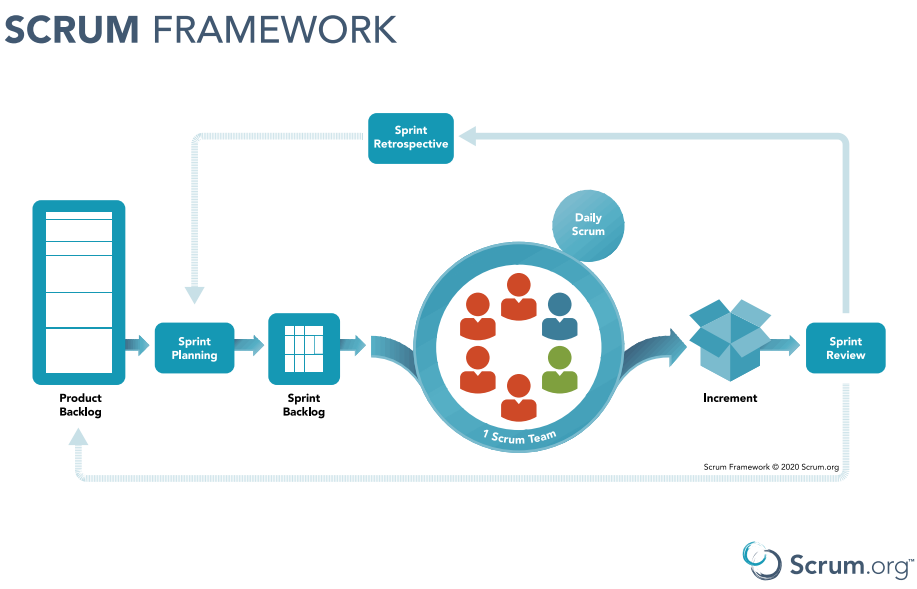


Figura 11 - Proceso SCRUM. Fuente: scrum.org [4]

Por otro lado, también se ha aplicado la metodología de *Kanban*, la cual consiste en uno o varios tableros en los cuales a través de columnas se distribuyen las tareas a realizar por parte del equipo de desarrollo permitiendo visualizar el flujo de trabajo, identificar cuellos de botella y mantener un seguimiento claro del progreso de las tareas según el estado en el que se encuentren. Cada vez que dicha tarea modifique su estado, esta se pivotara entre las distintas columnas. Por su parte también se suele indicar un límite de tiempo máximo para implementar dicha funcionalidad. Esto permite definir los costes de tiempo y evaluar si las estimaciones de trabajo se han cumplido o si por el contrario hay que mejorar durante el desarrollo del producto.

En cuanto al desarrollo específico de dicho proyecto, se aplicó el método de *Kanban* para reflejar en dicho tablero los casos de uso a implementar. La representación de cada columna que se ha utilizado es la siguiente:

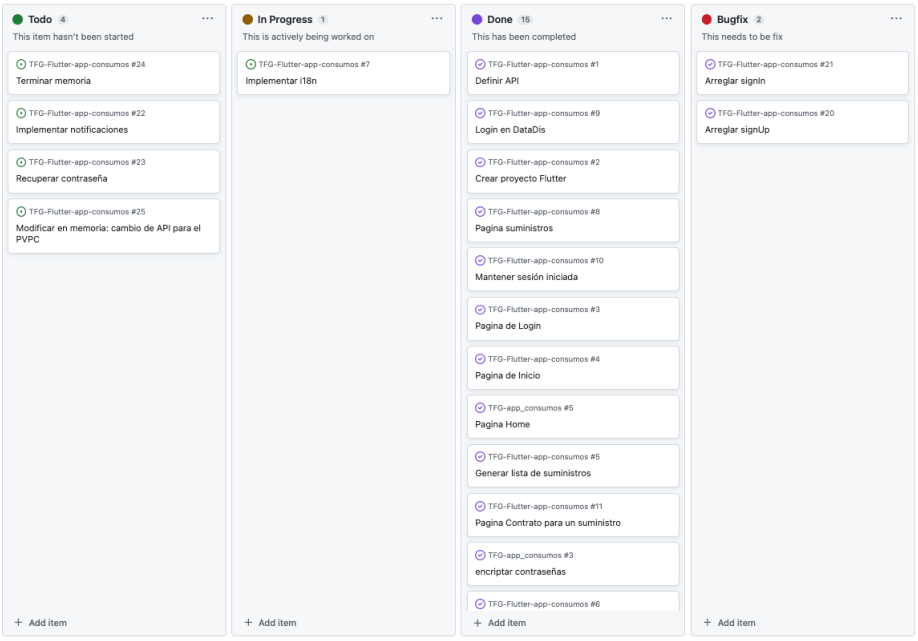


Figura 12 - Tablero Kanban

- **Todo:** Esta columna representa todas las tareas pendientes que deben realizarse. Aquí se incluyen todas las tareas que están en la lista de pendientes y aún no se han asignado a un miembro del equipo.
- **In Progress:** Una vez que un miembro del equipo selecciona una tarea de la columna "Todo", se mueve a la columna "In progress". Esto indica que la tarea está siendo trabajada actualmente. Puede haber varias tareas en esta columna al mismo tiempo, pero cada miembro del equipo generalmente se enfoca en una tarea a la vez para maximizar la eficiencia.
- **Done:** Cuando un miembro del equipo completa una tarea de la columna "In Progress", se mueve a la columna "Done". Esto indica que la tarea ha sido terminada y no requiere más trabajo. La columna "Done" muestra las tareas que se han completado exitosamente.
- **Bugfix:** Si se detecta un error en una tarea que se consideró "Done", se puede mover a la columna "Bugfix" para que se aborde y se realice la corrección necesaria. Una vez que se haya solucionado el error, la tarea se puede devolver a la columna "In progress" para su verificación y luego mover nuevamente a "Done".

Ambas metodologías pueden aplicarse flexiblemente y adaptadas a las necesidades específicas del desarrollador en este proyecto. Lo importante es establecer una estructura y rutina para el trabajo, establecer objetivos y hacer un seguimiento continuo para garantizar el éxito del proyecto.

Las principales ventajas entre aplicar metodologías ágiles comparado con las tradicionales son principalmente una mayor flexibilidad a los cambios en los requisitos del proyecto, apartar una entrega continua al cliente, mejorar la comunicación entre el equipo de desarrollo y el “Product Owner”, una mayor involucración de este con el producto y permitir detectar de forma temprana los problemas y ahorrar tiempo y costes a largo plazo.

4.2 Planificación

Durante la planificación inicial durante el desarrollo del *TFT 02* se planeó el siguiente esquema:

Fases	***Duración Estimada***	***Tareas***
Estudio previo / Análisis	50	Análisis de requisitos
		Estudio herramienta Flutter
Diseño / Desarrollo /Implementación	190	Diseño de las funcionalidades
		Implementación de las operaciones
		Creación de las funcionalidades
		Implementación de la visualización de los resultados
Evaluación / Validación / Prueba	20	Validación de las funciones en el lado cliente
		Evaluación del acceso a la API de Datadis
Documentación / Presentación	40	Redacción de la memoria
		Preparación de la presentación

Tabla 2 - Planificación Inicial

Durante el desarrollo del proyecto, se pudo ir comprobando que finalmente la planificación inicial propuesta no se adaptó de manera ajustada al desarrollo. Ya que hubo fases en las que el tiempo invertido fue mayor, a continuación de detalla en otra tabla el total dedicado a sección de manera definitiva:

Fases	*Duración Resultante*	*Tareas*
Estudio previo / Análisis	62 	Análisis de requisitos
		Estudio herramienta Flutter
Diseño / Desarrollo /Implementación	192	Diseño de las funcionalidades
		Implementación de las operaciones
		Creación de las funcionalidades
		Implementación de la visualización de los resultados
Evaluación / Validación / Prueba	6 	Validación de las funciones en el lado cliente
		Evaluación del acceso a la API de Datadis
		Evaluación del acceso a la API de Red Eléctrica
Documentación / Presentación	40	Redacción de la memoria
		Preparación de la presentación

Tabla 3 - Planificación Final

Dicho desajuste se debe principalmente, puesto a que resultó que, durante el análisis de las **API** a consumir, su documentación estaba bastante bien explicada y sencilla de utilizar. En cambio, se le dedicó un mayor número de horas al análisis y estudio de la herramienta de **Flutter**.

Por otro lado, también destacar que, durante el desarrollo del mismo, en la API de **Datadis** hubo cambios en su método de utilización durante el transcurso del desarrollo, por lo que hubo que realizar modificaciones oportunas, las cuales se incluyen de igual manera en el apartado de validación y prueba.

Indicar de igual manera, que gracias al uso de librerías o paquetes de **Flutter**, el desarrollo de ciertas funcionalidades se realizó de manera que hizo falta invertir un menor número de horas.

A la hora de la distribución de las tareas a realizar durante este proyecto al haber aplicado la metodología de *SCRUM* se ha implementado un "Product Backlog", con una serie de casos de uso e historias de usuario (HU), agrupados por "Sprints", donde cada uno de ellos han tenido la duración de 2 semanas aproximadamente.

Historias de usuario (HU)

Durante esta fase, se crearon diversas historias de usuario que delimitaban las funcionalidades de la plataforma. Estas historias de usuario, como se mencionó anteriormente, son descripciones breves que resumen las necesidades de los usuarios al utilizar el servicio.

El esquema de dichas HU es el siguiente:

- **Identificador**: conformado por el acrónimo y el número de la historia (HUxx).
- **Título de la historia**
- **Rol**: indica el tipo de usuario que realiza la acción.
- **Quiero**: acción que se desea alcanzar.
- **Para**: objetivo de la historia de usuario.
- **Criterio de aceptación**: criterio que se debe cumplir para dar por terminada la historia de usuario.

A continuación, se incluirán algunas las distintas historias de usuario más relevantes implementadas a lo largo de este trabajo, en el Anexo II se podrán visualizar el resto de las historias de usuario que no se hayan descrito en este apartado.

ID	HU01
TITULO	Registro de usuario
ROL	Usuario no registrado
QUIERO	Crear una cuenta en la plataforma
PARA	Acceder a las funcionalidades de la aplicación
CRITERIO DE ACEPTACION	- Debe estar previamente registrado en la plataforma de Datadis. - Debe existir una página dedicada al registro de usuarios. - El formulario incluirá entre los datos: email, nombre, apellidos, contraseña, NIF y contraseña de la plataforma usada en Datadis. - Ambas contraseñas deberán ser encriptadas usando una clave HASH. - Se deberá comprobar que el usuario no este previamente registrado en la base de datos. - Una vez registrado, se redirigirá al usuario a la página principal de la aplicación.

Tabla 4 - HU01 Registro de usuario

ID	HU02
TITULO	Iniciar sesión
ROL	Usuario registrado
QUIERO	Iniciar sesión en la plataforma
PARA	Acceder a las funcionalidades y a mis datos
CRITERIO DE ACEPTACION	- Debe estar previamente registrado en la plataforma. - Debe existir una página dedicada para iniciar sesión de usuarios. - El formulario incluirá entre los datos: email y contraseña. - La contraseña se descifrará de la base de datos para ver si coincide. - Una vez autentificado, se redirigirá al usuario a la página principal de la aplicación.

Tabla 5 - HU02 Iniciar sesión

ID	HU05
TITULO	Visualizar PVPC
ROL	Usuario registrado
QUIERO	Visualizar en la pantalla principal el precio medio eléctrico del día actual
PARA	Poder saber el precio en cada momento del día
CRITERIO DE ACEPTACION	- Debe estar autentificado en la aplicación. - Se mostrará en pantalla tres elementos con el precio mínimo del día, el actual y el máximo. - Se mostrará un gráfico visual con los precios por hora a lo largo del día.

Tabla 6 - HU05 Visualizar PVPC

ID	HU06
TITULO	Consumo para un suministro
ROL	Usuario registrado
QUIERO	Visualizar el consumo de un suministro
PARA	Poder consultar el consumo en un día o rango de fechas
CRITERIO DE ACEPTACION	- Debe estar autentificado en la aplicación. - Se mostrará un calendario, del que se podrá seleccionar los elementos a consultar. - En caso de tener elegido el modo día: se generará un gráfico de barras en una nueva ventana con los consumos por hora en kW. - En caso de tener elegido el modo rango: se generará un gráfico de barras en una nueva ventana con los consumos totales por día en kW.

Tabla 7 - HU06 Consumo para suministro

ID	HU03
TITULO	Encriptar contraseñas
ROL	Administrador
QUIERO	Asegurar las contraseñas de los usuarios
PARA	Asegurar la información de los usuarios
CRITERIO DE ACEPTACION	- Se creará una clave HASH única para encriptar los datos (no puede ser compartida y alojarse en un fichero externo). - Cualquier dato sensible como contraseñas deberá pasar por el método `encrypt()` antes de ser alojado en la base de datos. - Al leerse de BD deberá pasar por el método `decrypt()`.

Tabla 8 - HU03 Encriptar contraseñas

ID	HU12
TITULO	Comparar consumo para un suministro
ROL	Usuario registrado
QUIERO	Comparar el consumo generado con otros
PARA	Poder comparar mi consumo con los de otros usuarios
CRITERIO DE ACEPTACION	- Debe estar autentificado en la aplicación. - Se deberá previamente consultar el consumo de un suministro. - Se mostrará dos ítems desplegables uno para provincias que será obligatorio y uno filtrado por el anterior donde se muestren los municipios, el cual será voluntario su elección. - El resultado será una lista donde se mostrará el promedio del consumo en la zona a comparar y al lado el del suministro consultado, indicado con colores si el consumo es mayor (rojo) o menor (verde).

Tabla 9 - HU12 Comparar consumo

ID	HU07
TITULO	Implementación i18n
ROL	Administrador
QUIERO	Visualizar la aplicación en distintos idiomas
PARA	Poder ver la aplicación en otro idioma si no hablo español
CRITERIO DE ACEPTACION	- En la pantalla de autenticación se mostrará en la parte superior un icono que desplegará un menú con los idiomas disponibles. - En la pantalla principal o home se mostrará de igual manera un icono para cambiar el idioma. - Por defecto el idioma seleccionado será español, salvo que se indique lo contrario. - Al cambiar el idioma, se actualizará la aplicación con sus contenidos en el idioma seleccionado.

Tabla 10 - HU07 Multilenguaje

Sprint - 0

En este primer “Sprint” inicial, se ha definido como objetivo la configuración inicial del proyecto y la preparación del entorno del trabajo.

- Crear proyecto Flutter.
- Investigación sobre librerías.
- Investigar sobre la API de Datadis.
- Investigar sobre la elección de base de datos.

Sprint - 1

En este primer “Sprint” inicial, se ha definido como objetivo el tener la aplicación inicial ya operativa y el poder disponer de un mínimo de operaciones funcionales a realizar por el usuario. Entre las historias de usuario planteada se encuentran:

- Operar con la app de Datadis para hacer consultas a la API con datos estáticos.
- Crear primera página de inicio o pantalla principal.
- Establecer un diseño para dicha pantalla.
- Crear primera página de inicio de sesión y registro.
- Modelar la base de datos.

Dentro de este primer “Sprint” no hubo problema para realizar cada una de las tareas que se tenían planteadas, por lo tanto, la estimación de esfuerzo fue considerada en corto y para las próximas se tuvo en cuenta el asignar más tareas al mismo.

Sprint - 2

En este “Sprint”, se define como objetivo el “refactorizar” una serie de pasos desarrollados en la etapa anterior, así como añadir nuevas funcionalidades a la misma y además ya poder disponer de datos dinámicos. Entre las historias de usuario planteada se encuentran:

- Aplicar librería de *flutter_login* para sustituir la actual pantalla y sistema de inicio de sesión.
- Investigación sobre librería de *flutter_login.*
- Crear todos los puntos de acceso a la API y sus llamadas de Datadis.
- Crear página inicial de desplegable de suministros.
- Crear página de perfil del usuario para poder cerrar sesión.
- Nutrir a la base de datos con datos creados de los usuarios.

En esta fase se detectaron problemas en cuanto a la gestión de los “tickets” puesto que el volver a readaptar la pantalla de inicio de sesión a una librería ya definida con una estructura, hubo que investigar en detalle la misma. Además, al haber realizado un cambio de base datos se complicó la nueva adaptación a los métodos de lectura y escritura de los mismos.

Sprint - 3

En este “Sprint”, es donde más cambios y tareas se desarrollaron además de tener más soltura con las herramientas utilizadas. Se ha definido como objetivo el disponer de una versión funcional ya operativa con las principales funcionalidades finales a falta de diseño o correcciones. Entre las historias de usuario planteada se encuentran:

- Crear página de visualización del consumo de un suministro.
- Crear página visualización de contrato de un suministro.
- Refactorizar página desplegable de los suministros, incluyendo botones para consultar las páginas anteriores.
- Refactorizar página principal, donde ahora se mostrará un panel de control sobre el PVPC del día actual.
- Crear página de comparación de suministros una vez consultas el consumo de un suministro.

Dentro de este “Sprint” hubo una gran carga de tareas, y finalmente la duración del Sprint tuvo que ampliarse a una semana más para poder finalizar las tareas dentro del mismo. Finalmente, y gracias a ampliar dicho plazo se consiguió realizar las tareas que se tenían previstas.

Sprint - 4

En este penúltimo “Sprint”, se ha definido como objetivo el dotar de un diseño uniforme a lo largo del transcurso de navegación entre las distintas pantallas. Entre las historias de usuario planteada se encuentran:

- Dar un diseño uniforme a cada página de la aplicación.
- Crear un logo para la aplicación.
- Refactorización de código de “Sprint” anteriores.
- Investigar sobre notificaciones “push”.

En este “Sprint” se pudo desarrollar la mayoría de las tareas, excepto la implementación de las notificaciones, puesto que para ello haría falta de algún gestor externo.

Sprint - 5

En la última fase se ha dedicado a la refactorización general y revisión de los diferentes casos de uso y comprobar su validación.

4.3 Presupuesto

A la hora de establecer un presupuesto para el coste de una herramienta de dichas características, puede parecer complejo ya que en un proyecto real dicho trabajo sería realizado por un mayor número de personas además de los costes podrían verse influenciados por distintos factores.

Para el caso de este proyecto se ha supuesto que todo el desarrollo ha sido realizado por una única persona utilizando en su mayoría servicios gratuitos, que en caso de resultar en una escalabilidad mayor haría falta recurrir a servicios de pago para alojar los datos e incluir nuevas funcionalidades a la aplicación, lo cual podría además repercutir en un segundo desarrollador.

Un presupuesto aproximado para este proyecto a fecha de hoy sería el siguiente:

Hardware	
MacBook Pro 15" 2013 (Reacondicionado)	549,00 €
Software	
Visual Studio Code	0,00 €
Postman	0,00 €
Base de datos SQLite en local	0,00 €
Diagrams.net	0,00 €
Despliegue	
Google Play Store	22,00 €
Apple Store	100,00 €
Amazon	0,00 €
Personal	
Desarrollador full stack	5400,00 € (17,95 €/hora)
Total	**6049,00 €**

Tabla 11 – Presupuesto del trabajo

Dichos cálculos han sido estimados consultando los precios a fecha del día de la redacción de dicha tabla, en su mayoría al ser herramientas gratuitas no suponen ningún costo.

En cuanto al equipo hardware utilizado, si bien es una unidad antigua aún dispone de unas características capaces para llevar a cabo tareas de desarrollo como las de este proyecto. Su precio por ende está calculado en base a modelos actuales disponibles los cuales ya no están en producción pero sí se pueden adquirir de forma reacondicionada o de segunda mano, tras un breve análisis a través de la web de **Amazon** se ha encontrado el mismo modelo con un precio de 549€.

Por otro lado, en cuanto, al costo del despliegue de las aplicaciones en los distintos *Marketplace* puede ser consultado a través de las propias páginas de los distribuidores. Cabe destacar que Amazon al ser relativamente nuevo y pequeño su coste es 0€, en cambio en los otros dos, se cobra una cuota inicial, además de un porcentaje de los ingresos a través de la aplicación.

Finalmente, en cuanto al salario del personal, para un promedio en España de un desarrollador "full stack", se ha calculado que es de entorno a los 35000€ al año o 17,95€/hora. Por lo tanto, para un proyecto de 300 horas, se obtiene un salario de 5400€. Dicha fuente ha sido obtenida a través de talent.com, un portal de estimaciones de salarios promedio para distintas áreas o sectores profesionales.

Capítulo 5 Herramientas Utilizadas

Seguidamente se procederá a exponer los recursos y herramientas utilizadas a lo largo del proyecto, incluyendo los lenguajes de programación, "frameworks", entornos de desarrollo, almacenamiento de datos u otras tecnologías empleadas.

5.1 Herramientas software

Para el desarrollo de este proyecto se ha empleado **Dart**, un lenguaje de programación desarrollado por Google en el año 2013, con el fin de poner solución a algunos problemas relacionados con el uso de JavaScript. **Dart** se utiliza principalmente como lenguaje de programación para desarrollar aplicaciones móviles y web. Una de las principales razones por las que **Dart** es popular es su estrecha relación con **Flutter**.

Flutter, por su lado, es un "framework" de código abierto también desarrollado por Google. Utiliza el lenguaje **Dart** como su lenguaje de programación principal. **Flutter** permite crear aplicaciones móviles de alta calidad de forma rápida y eficiente, utilizando una sola base de código para iOS y Android.

La relación entre **Dart** y **Flutter** es muy estrecha, ya que **Dart** es el lenguaje utilizado para escribir la lógica de la aplicación en **Flutter**. **Dart** proporciona características como tipado estático, recolector de basura, programación asíncrona y programación orientada a objetos, que son fundamentales para el desarrollo de aplicaciones en **Flutter**.

En resumen, **Dart** es el lenguaje de programación utilizado en **Flutter**, y la combinación de ambos permite a los desarrolladores crear aplicaciones móviles de alto rendimiento y con una interfaz de usuario atractiva.

Figura 13 - Logos de Dart y Flutter. Fuente: inLab [5]

Frameworks

Para permitir un uso más sencillo y eficaz de la creación de las interfaces de usuario de la aplicación se ha utilizado el *SDK*[5] propio de Google creado para **Dart**, **Flutter**. Dicho "framework" ha notado un aumento notable de popularidad entre el sector de desarrolladores debido entre otras cosas a su notable velocidad de nuevas funcionalidades incluidas, una experiencia nativa y construcción de elementos de interfaces sencillo.

Su principal ventaja es que, incluye la incorporación de "*Widgets"*. Dichos elementos son componentes que permiten construir toda la interfaz desde elementos sencillos como botones o texto, hasta elementos más complejos como gráficos, formas y animaciones.

En el caso de no disponer de los "*widgets"* necesarios para su construcción **Flutter** permite la creación y combinación de elementos más sencillo.

Librerías

Además de las funciones principales que ofrece el "framework", **Flutter** también incluye una amplia variedad de librerías y/o paquetes que pueden ser utilizados por los desarrolladores para facilitar aún más la construcción de componentes de la interfaz de usuario y funcionalidades a implementar en la aplicación.

Estas librerías o paquetes pueden ser consultados en el enlace pub.dev. Utilizar estos paquetes puede ahorrar tiempo y esfuerzo en el desarrollo, ya que ofrecen una funcionalidad predefinida que puede ser utilizada sin necesidad de escribir el código desde cero, y pueden permitir a los desarrolladores implementar características más avanzadas sin necesidad de una formación más profunda.

Entorno de desarrollo

Se ha utilizado mayoritariamente el entorno de desarrollo de *Visual Studio Code*. Pues dicho entorno además de ser gratuito y estar perfectamente adaptado para el desarrollo mediante **Flutter**, permite también un gran soporte de diversas extensiones alguna de ellas útiles para simplificar el desarrollo, como alguna de ellas meras visuales que favorecen al aspecto visual del mismo, repercutiendo de esa manera en una mayor concentración.

Figura 14 - Logo Visual Studio Code. Fuente: Canonical Snapcraft [6]

[5] SDK: "Software Development Kit" es un conjunto de herramientas de software que permite crear aplicaciones.

5.2 Herramientas de apoyo

Entre las herramientas utilizadas durante el desarrollo del proyecto se emplearon, estas se dividen entre aquellas que nos valen para el control de versiones y gestión del proyecto, y el resto que han sido un pilar de apoyo durante el desarrollo, para facilitar algunas tareas.

Control de versiones y proyecto

Dentro de este apartado encontramos las herramientas de **GitHub**, utilizado como repositorio para mantener un control de versiones. Por otro lado, se ha utilizado una herramienta que aporta **GitHub** también para la gestión de los propios proyectos **GitHub Project**, esta herramienta se utilizó para implementar la metodología utilizada mencionada en el capítulo anterior.

Herramientas de proyectos

Postman es una herramienta que sirve para la realización de pruebas mediante solicitudes HTTP a través de la conexión a una API. Esta se ha empleado para las pruebas pertinentes de la API desarrollada para la plataforma, ya que permite probar los "*endpoints"* del servidor sin la necesidad de programarlas previamente, lo que facilita su implementación más tarde.

También se utilizó **DB Browser**, un programa gratuito para el sistema *MacOS*, utilizado para permitir la visualización de los datos almacenados en la base de datos de manera más directa al momento de insertar o actualizar elementos en la misma.

Por otro lado, se utilizó también el **Simulator**, una aplicación nativa en sistemas *MacOS*, que permite cargar un simulador de cualquier dispositivo Apple para poder realizar las pruebas en entorno simulado de un dispositivo.

5.3 Base de datos

En cuanto al gestor de base de datos utilizado inicialmente al comienzo del proyecto, **Firebase**, se encontró que su uso afectaba negativamente los tiempos de inicialización de la aplicación. Además, la manipulación de datos implicaba un alto costo en comparación con el sistema finalmente seleccionado.

Después de evaluar las opciones disponibles, se decidió utilizar un motor derivado del lenguaje de **SQL**, específicamente **SQLite**. Esta elección se basó en la capacidad del mismo para realizar transacciones pequeñas, rápidas, seguras y altamente confiables, todo ello con un costo relativamente bajo. Dada la falta de necesidad de escalabilidad en el período de desarrollo del proyecto, **SQLite** resultó ser la opción ideal.

5.4 Sistema operativo

El sistema operativo usado es *MacOS* en su versión 12 debido a que es el ordenador personal que disponía durante el desarrollo.

En cuanto a las pruebas de la aplicación al ser desarrollada para plataformas móviles, las pruebas se realizaron con simuladores iOS y Android a través de las herramientas mencionadas anteriormente.

Además, se ha utilizado un dispositivo físico iPhone 8 Plus, para ver cómo sería la fase final ya llevada a la fase de producción con dispositivos físicos reales.

Capítulo 6 Desarrollo del Proyecto

A lo largo de este apartado se procederá a explicar las etapas, planificación, diseño e implementación llevados a cabo durante el desarrollo del proyecto. Concretamente se hablarán sobre los casos de uso, sus actores implicados, acerca del diseño de la aplicación y la arquitectura de la misma.

6.1 Análisis de requisitos del software

En esta primera etapa, se hará énfasis en los requisitos funcionales y no funcionales, es decir las historias de usuario y casos de uso.

Actores implicados

Durante el flujo de la aplicación intervienen una serie de actores a lo largo del mismo. Entre estos se hallan: el usuario no registrado, el usuario registrado, y luego por otra parte de manera indirecta intervienen **Datadis** y **REE** a través de la conexión a sus API.

Usuario no registrado: usuario por defecto el cual entra por primera vez en la aplicación y sus funciones están muy limitadas.

Usuario registrado: usuario estándar que puede acceder a las funcionalidades completas del aplicativo.

Datadis: actor indirecto, el cual actúa en una serie de funcionalidades ofreciendo los servicios de acceso a sus distintos puntos de la **API privada**.

REE: actor indirecto, el cual actúa en la funcionalidad del ofrecer el precio del PVPC a través de uno de sus puntos de la **API pública**.

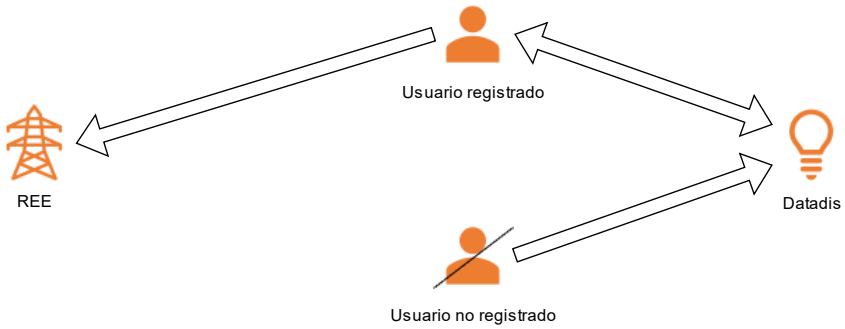


Figura 15 - Diagrama interacción entre usuarios

Casos de uso

En dicho apartado, se muestra de manera gráfica y esquemática las tareas y actividades que un actor realiza durante el flujo de uso de la aplicación.

A la hora de realización de dichos diagramas hay que considerar entre otras cosas, quienes son los actores que interactúan con la aplicación, es decir los roles que existen dentro de la aplicación. Estos reciben también el nombre de actores, en este proyecto se identifican los siguientes roles:

- **Usuarios no registrados**:
  - **Registro en sistema**: para ello rellenará un formulario de acceso, el cual será verificado por Datadis y en caso de procederse a la creación del usuario significa que el mismo está registrado a través de la plataforma.
- **Usuarios registrados**:
  - **Consultar el PVPC**: para el cual intervendrá el actor REE, del cual tomamos la acción de traer a través de la API del mismo, los datos del día actual en cuanto al precio de la electricidad.
  - **Consultar consumo**: A la hora de consultar el consumo puede o no intervenir el actor Datadis, dependiendo si es la primera vez que se realiza una consulta para dicho consumo, es decir, en caso de ya haberse realizado una consulta previa sobre el mismo mes, este no intervendrá, en caso contrario, se traerá a través de la API de Datadis los datos necesarios que serán posteriormente alojados en la BBDD.
  - **Consultar contrato**: de igual manera, en caso de ser la primera visualización del mismo, intervendrá el actor Datadis.
- **Datadis**: actor intermediario en una serie de operaciones como pueden ser la verificación de usuario registrado, autenticación y generación de clave de acceso, descarga de datos relacionado con los suministros.
- **REE:** actor intermediario, que permite obtener los precios actuales de la electricidad para el día de hoy.

Diferenciando los actores descritos anteriormente, es importante también visualizar las acciones que realizan dichos actores. Por lo que se han generado dos esquemas uno para cada tipo de usuario en la aplicación y sus actores intermedios. En el primero de ellos (Figura 16) observamos los distintos casos de uso disponibles para un usuario que aún no se ha registrado en la plataforma y que solo realiza una consulta a Datadis.

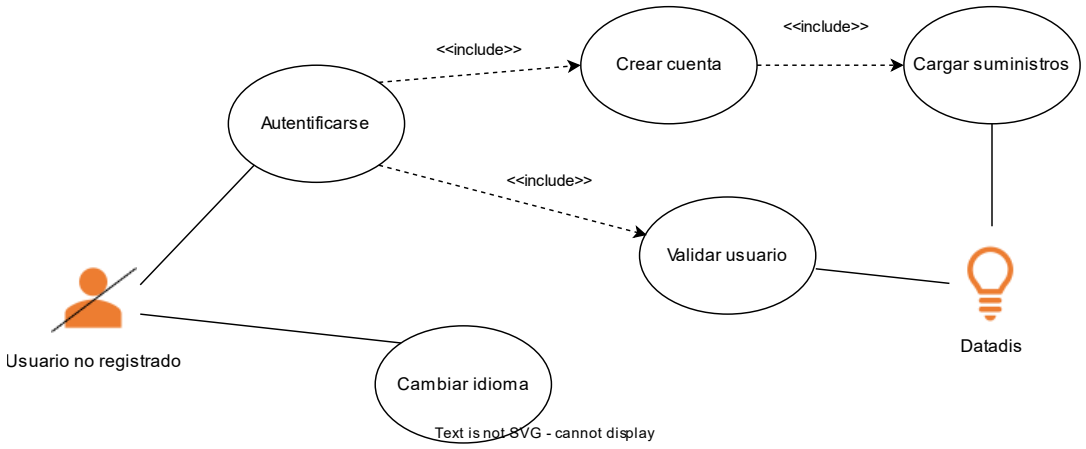


Figura 16 - Caso de uso: Usuario no registrado

Por otro lado, se ha generado el siguiente diagrama para el caso de uso, donde el usuario está registrado. En el mismo se pueden distinguir dos ramas conjuntas. La primera obtiene conexión a través de REE (Figura 18). De otro lado tenemos la cual realiza la conexión con Datadis (Figura 17) para aquellas tareas relacionadas con el uso de los suministros y sus consultas.

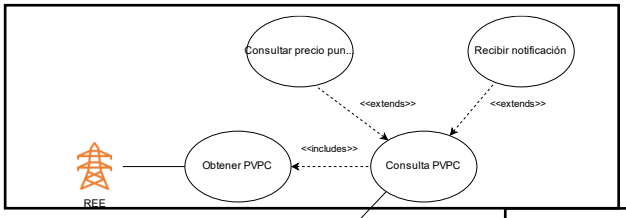


Figura 18 – Casos de uso a través de REE

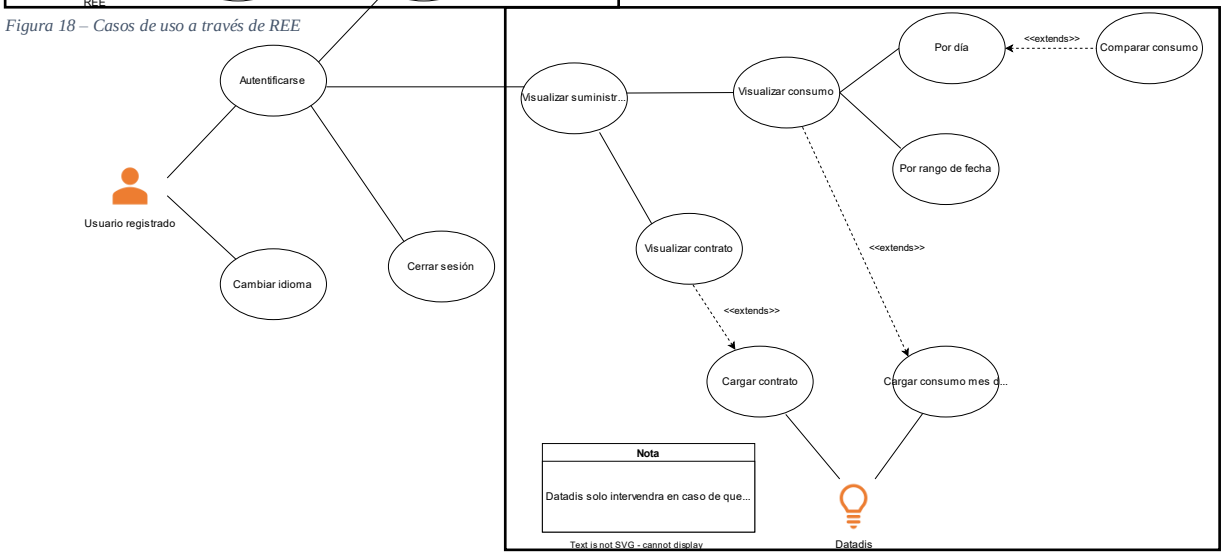


Figura 17 – Casos de uso a través de Datadis

Figura 19 - Caso de uso: Usuario registrado

6.2 Diseño

En este apartado se hablará de la gestión y almacenamiento de los datos y cómo se tratan los mismos. En este es donde almacenarán los datos de los usuarios y los suministros asignados a los mismos. Por otro lado, también se almacenarán los consumos generados, para tener un acceso más rápido a los mismos, y poder compararlos con los de otros usuarios.

Todos los datos recibidos serán tratados en formato *JSON*[6], por lo que habrá que convertir dichas cadenas a objetos propios con un contexto dentro de la aplicación. Todos estos datos se tratarán en una misma clase *´API.dart´* y luego en las distintas vistas o pantallas será donde se haga una llamada a dicha clase para hacer uso de los objetos.

Diseño de la arquitectura del sistema de la base de datos

La base de datos del sistema es un elemento clave para el correcto funcionamiento de este, siendo esta la razón por la cual un buen diseño mejora y facilita la posibilidad de futuras actualizaciones y mantenimientos.

Con esto en mente, inicialmente se decidió su implementación mediante **Firebase Database** una base de datos **NoSQL** documental desarrollada por Google. Dicha opción tras un previo análisis de los datos a tratar y una primera implementación de los datos, se llegó a la conclusión de que no resultaba óptima para el almacenamiento de los datos ya que los datos a tratar son únicamente estructurados. Por otro lado, se pudo ver que la carga inicial de la app se demoraba mucho en tiempos de ejecución debido a la cantidad de paquetes que necesita **Firebase** para su funcionamiento. Por ello se optó finalmente por otro sistema para la gestión de la base de datos.

Tras un análisis posterior de las distintas opciones del mercado disponible, se decidió optar por **SQLite** cuyas ventajas ya se han explicado a lo largo de este mismo documento.

En cuanto al modelado del diseño y su arquitectura, se ha optado por el modelo de **arquitectura cliente-servidor**, donde se identifican claramente dos partes:

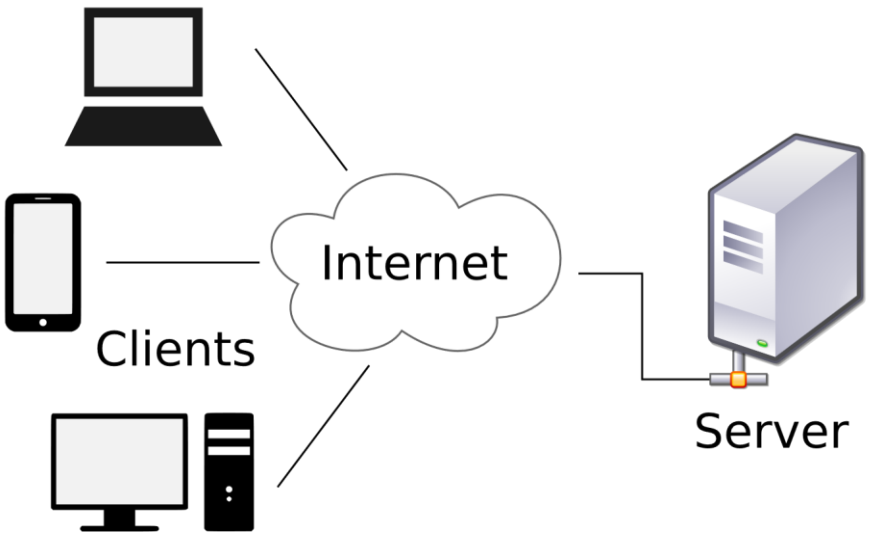


Figura 20 - Arquitectura cliente-servidor. Fuente: Wikipedia [7]

[6] JSON: formato de texto sencillo para el intercambio de datos. [30]

Por un lado, el servidor, donde se alojará una base de datos **SQLite** que contienen la información de los consumos eléctricos de los usuarios, así como la información relacionada de los mismos. Esta base de datos se encargará de almacenar y gestionar los datos de consumo, como los registros de consumo de energía, los datos del usuario, etc.

Esta capa a su vez también cuenta con una capa lógica de negocio que maneja las solicitudes y consultas de los clientes, interactuando con la base de datos para recuperar y actualizar los datos según sea necesario.

Por otro lado, el cliente será la aplicación móvil propiamente instalada en el dispositivo.

Cuando un usuario abre la aplicación móvil, se establece una conexión con el servidor a través de la red. El cliente envía solicitudes al servidor para recuperar los datos de consumo eléctrico del usuario específico.

El servidor procesa la solicitud del cliente, consulta la base de datos **SQLite** para obtener los datos solicitados y luego envía la respuesta al cliente. El cliente recibe los datos y los muestra al usuario de forma legible, como gráficos, tablas u otras representaciones visuales.

Este modelo resulta ideal, cuando se quiere distribuir cada capa con responsabilidades distintas. Además, destacar que la opción de optar por **SQLite** al ser una base de datos de servidor liviana es adecuada para aplicaciones móviles con una carga de trabajo moderada. Sin embargo, si se espera un alto volumen de usuarios o una mayor complejidad en los datos y consultas, podría ser necesario considerar otras opciones de base de datos más robustas.

Finalmente remarcar si bien dicha arquitectura lo más común es que esté pensada para que el servidor esté alojado en una localización centralizada o en la nube y el cliente se comunique con el servidor, para el caso de uso de este proyecto se decidió implementar en local la base de datos directamente también en el dispositivo del usuario.

Como se indica en el apartado de Conclusiones, lo ideal en un trabajo próximo a la hora de llevar a despliegue este proyecto será trasladar el servidor a un entorno de red centralizado para que sea accesible a múltiples clientes y en caso de escalar muy rápidamente, la posibilidad de cambiar el gestor del sistema por uno que permita un mayor tráfico de consultas al mismo tiempo.

En cuanto a la estructura elegida para esta base de datos se ha optado por un “schema” el cual cuenta con 5 tablas relacionales, las cuales se pueden visualizar en la siguiente ilustración (Figura 21 - Diagrama ER) y se expondrá la función y campos de cada una de las mismas.

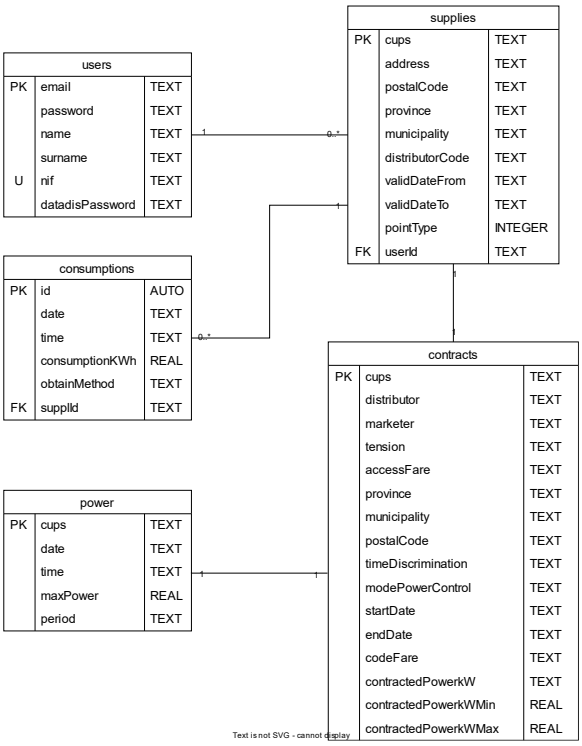


Figura 21 - Diagrama ER

- **users**: corresponde a la información relacionada con los usuarios, y se relaciona con los suministros. Donde un usuario puede disponer de ningún o varios suministros a su nombre.
- **supplies**: información relacionada con los suministros tales como, cups e información relacionada con el punto del suministro.
- **contracts**: es la tabla que contiene mayor número de columnas o información al respeto. En dicha página se detallan la información del punto del suministro.
- **consumptions**: información donde se alojan los consumos de los suministros registrados en la base de datos. Entre sus columnas encontramos los siguientes campos:
    - **cups**: número de identificador único.
    - **date**: día del consumo del suministro en formato
    - **time**: hora del consumo.
    - **consumptionKwh**: valor del consumo para dicho punto.
- **powers**: aloja la información relacionada con las potencias eléctricas de un suministro.

Entrando más en detalle en la implementación de la base de datos, se ha creado una clase 'DB.dart' la cual gestiona las operaciones de creación de tablas, inserción de datos, lectura de datos, y borrado de datos.

El primer paso que se lleva a cabo en transcurso de vida de la aplicación es la apertura de la base de datos, para ello en el método `*main()*` de la misma, debemos insertar la operación de inicialización de apertura de la base de datos `*openDB()*`. El código mencionado se puede consultar a través del Anexo IV -Código fuente relacionado con operaciones de la base de datos.

Entrando en el método realizaremos la apertura de la base de datos y en caso de no estar creadas las tablas por defecto, procederemos a crearlas usando la sintaxis propia de **SQLite**.

Tras la apertura de la base de datos, esto implica que la misma ya está cargada lo que significa, que se pueden ejecutar consultas y realizar modificaciones en los datos almacenados en la base de datos. Esto implica que se pueden realizar acciones como la inserción, actualización y eliminación de registros, así como la realización de consultas para recuperar información específica.

A partir de entonces podremos realizar las operaciones de lectura sobre la base de datos y las escrituras sobre la misma. Para demostrar el uso del mismo se expondrá a continuación, un ejemplo de consulta del contrato de un suministro y en caso de no existir en la base de datos se insertará en la misma.

En dicho ejemplo Anexo IV -Código fuente relacionado con operaciones de la base de datos, cómo se comentó, lo primero que haremos será abrir la base de datos para poder operar con la misa. Tras ello se intentará realizar una operación de lectura sobre la base de datos, invocando al método `*query()*` con los siguientes parámetros:

- tabla a consultar.
- columnas a retornar.
- condición o condiciones a cumplir.
- argumentos de las condiciones.

En el anterior ejemplo estamos retornando realizando una consulta a la base de datos de lectura para obtener una lista de contratos de los cuales nos quedaremos con el elemento primero, el cual contendrá la información relativa a un contrato.

En caso de que la consulta retornará un valor vacío, procederemos a la obtención de los datos a través de una llamada a la API usando para ello el valor del identificador (CUPS) y el código de distribuidora. Tras castear dicha respuesta JSON, procedemos a la inserción del elemento en la base de datos, para ello usaremos el método `*insert()*`.

En el ejemplo anterior hemos invocado al método `*insert()*` introduciendo como parámetros, el nombre de la tabla donde insertar los datos y un diccionario con los valores a insertar.

6.3 Diseño de la interfaz de usuario

En este apartado se expondrá el desarrollo relacionado por parte del lado del cliente, es además donde más énfasis se aportará a dicho proyecto ya que dicho factor era el más relevante de cara al usuario final.

El diseño de capa de presentación, también conocido como “front end”, es esencial en el desarrollo de una aplicación móvil puesto que debido a la naturaleza del tamaño reducido de pantalla y a la usabilidad de la misma se tiene que poder, ya que, en medida de lo posible poder ser utilizada a través de una única mano, se diseña una interfaz de usuario (UI) y una apariencia visual e interacción se tiene que tener como objetivo final, que sean atractivas con el usuario.

Se debe tener en cuenta que esta aplicación esta importancia se ve reforzada en mayor medida si cabe, puesto que, el usuario objetivo al que se tiene pensado dirigir son personas de una edad elevada y no a personas jóvenes.

La consistencia, por su parte, es clave para brindar una experiencia de usuario coherente a lo largo del transcurso o “flow” de la aplicación. Se deben utilizar elementos consistentes para

que el usuario pueda de manera rápida y sencilla adaptarse con la interfaz y mejorar su usabilidad. Para ello se tiene en cuenta al tratarse de dispositivos móviles el uso de espacios en blanco a modo de separadores visuales, evitando así la congestión visual y permitiendo destacar a los elementos clave de la interfaz.

La jerarquía visual, logra mediante el uso de diferentes patrones de tamaños, colores y estilos de tipografía indicar la importancia y la relación de los elementos de la interfaz. Por ello destacar dichos elementos ayuda de igual manera a comprender mejor la información presentada.

La accesibilidad sería otro factor clave a tener en cuenta, es decir, asegurar en la medida de lo posible que el diseño de la interfaz sea accesible para todos los usuarios. Para lograr esto, se deben utilizar colores y contrastes adecuados, proporcionar opciones de accesibilidad y considerar la legibilidad de los textos.

Los patrones de colores también desempeñan un rol importante en esta capa. Al elegir una paleta de colores, se debe buscar una combinación coherente que se ajuste a la identidad de marca y buscar que sea atractiva de igual manera. Para ello aplicamos la teoría del color, si bien este concepto puede llegar a ser muy complejo y abarcar muchos campos, en este contexto la referiremos como los principios y conceptos que nos ayudan a saber cómo interactúan los colores entre sí y como afectan nuestra percepción sobre los mismos.

Figura 22 - Esquema del círculo cromático [6]

En cuanto a su aplicación en el desarrollo, se han utilizado colores primarios para elementos clave de la interfaz, como botones principales o elementos de navegación destacados, y colores secundarios para resaltar elementos secundarios. También se han utilizado colores diferentes para representar diferentes estados de elementos como botones activos, elementos seleccionados, errores, o comparativas. Esto ayuda a los usuarios a comprender mejor el estado actual de la aplicación y facilita la interacción.

Finalmente, en cuanto a la experiencia de usuario (UX) es otro de los aspectos fundamentales en el diseño de una aplicación como esta. Como se comentó anteriormente debe ser intuitiva y fácil de usar, permitiendo con ello a los usuarios navegar y acceder a las funcionalidades sin dificultad.

Para ello también es importante diseñar flujos de navegación de manera concisa, aplicando estructuras lógicas y coherentes que permitan a los usuarios navegar sin esfuerzo alguno. En este caso se optó por utilizar un patrón muy común en este tipo de aplicaciones conocido como barra de navegación inferior. Dicha decisión se debió principalmente a la cantidad de categorías que realmente eran necesarias en este proyecto. Debido a la poca cantidad de elementos importantes a navegar se distribuyó de la siguiente manera, permitiendo siempre tener al alcance de un clic del usuario las distintas pestañas de navegación.

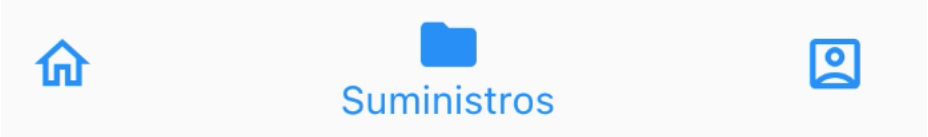


Figura 23 - Menú de navegación inferior

En base a todo lo anterior el uso de **Flutter** como "framework" de desarrollo, puede ser altamente beneficioso para implementar la capa de presentación ya que como se ha comentado previamente en este documento permite crear interfaces atractivas y de alto rendimiento de forma sencilla y eficaz.

Esto es posible gracias a los "Widgets", en base a su propia definición en la documentación oficial "Los widgets describen el aspecto que debería tener su vista en función de su configuración y estado actuales" [8], en propias palabras, son una serie de bloques que permiten construir elementos visuales, como botones, imágenes, tablas, etc. Estos elementos se utilizan para definir la estructura y la apariencia visual de la interfaz. Además, estos, pueden ser personalizados de manera que, si alguno de ellos no se adapta tal y como lo deseas puedes editarlo a tu gusto, o crear una variante del mismo.

Dentro de los mismos encontramos dos variantes:

- **Estructurales**: Son los responsables de establecer la jerarquía de las páginas y la organización de los elementos visuales que las componen. Algunos de estos elementos son:
    - MaterialApp
    - Scaffold
    - Container
    - SizedBox
    - Row
    - Column
    - Divider
    - Stack
- **Funcionales**: Aquellos que representan elementos visuales propiamente y a menudo son interactivos con el usuario. Algunos de estos widgets funcionales serían:
    - Text
    - ElevatedButton
    - Image
    - TextField
    - ListView

Cabe destacar además que estos elementos a subes se pueden anidar con otros elementos entre sí, para crear estructuras más complejas y que aporten una mayor funcionalidad.

Otra de las características más destacadas de **Flutter** es el "Hot Reload", que permite hacer cambios en el código y actualizar los cambios sin tener la necesidad de reiniciar completamente la aplicación. Esto agiliza el proceso de diseño y desarrollo, ya que permite ir probando variantes distintas sin invertir en ello mucho tiempo de espera, entre los cambios.

Por último, al ser multiplataforma, en base a un único código **Flutter** genera los códigos nativos para dispositivos iOS y Android. Esto implica una gran ventaja ya que, puedes utilizar los mismos diseños y componentes visuales en todas las plataformas, lo que garantiza una experiencia de usuario coherente y reduce el esfuerzo de desarrollo.

De forma resumida, con su amplio conjunto de componentes personalizables, capacidades de personalización visual y compatibilidad multiplataforma, **Flutter** permite crear interfaces de usuario atractivas y funcionales de manera eficiente.

Iniciar sesión y crear cuenta de usuario

La primera pantalla que se muestra al usuario será la de acceso a la aplicación, dentro de esta encontramos una serie de formularios en función de la acción que queramos realizar, iniciar sesión en una cuenta existente, o crear una nueva cuenta.

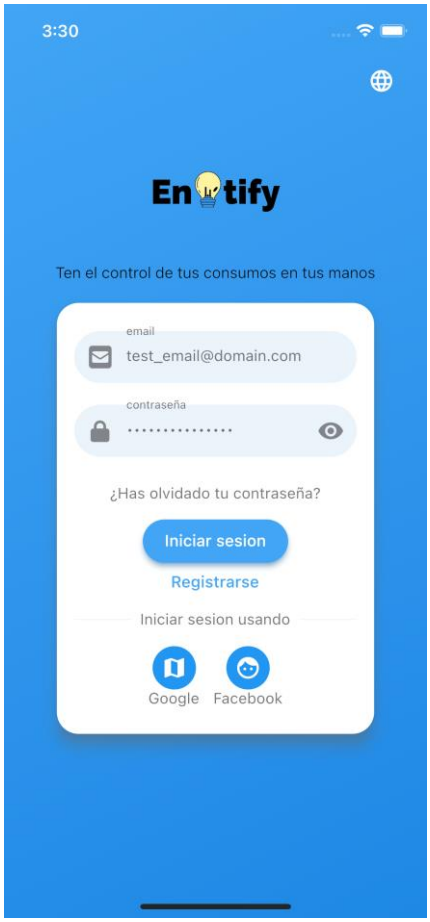


Figura 24 - Formulario de inicio de sesión

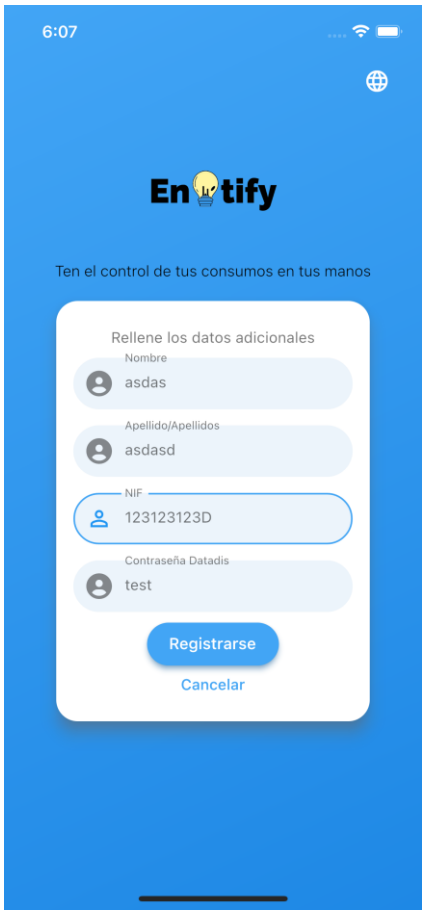


Figura 25 - Formulario de registro

Visualización del PVPC

Al iniciar la aplicación una vez de la autentificación o ya sea tras realizar el registro la primera página a mostrar al usuario será la página principal o Inicio, donde se mostrará una pantalla que contendrá el menú de navegación horizontal superior e inferior, además del resumen relacionado con el PVPC[7] para el día actual.

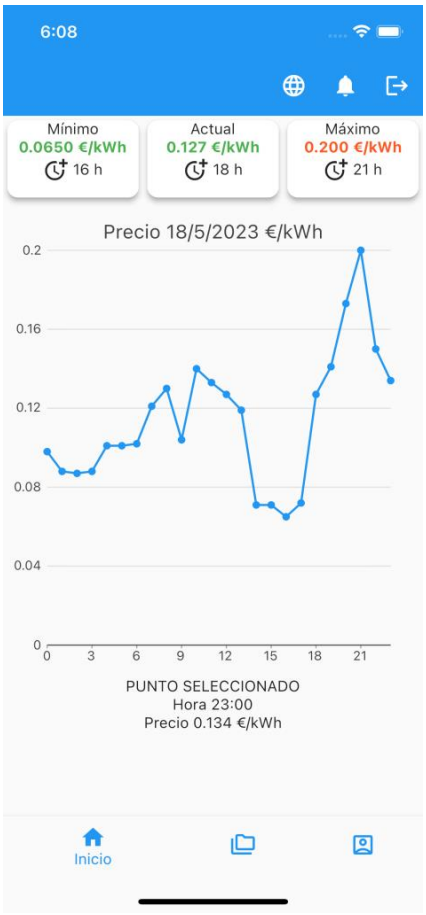


Figura 26 - Página de Inicio

En cuanto a los menús de navegación, en el primero de ellos se tendrá accesos directos, el primero de ellos permitirá cambiar la aplicación de idioma, un acceso a notificaciones y uno para permitir cerrar sesión (Figura 27 - Menú superior). En cuanto al menú de la parte inferior, mostrará los accesos directos a las funcionalidades principales de la aplicación: consultar página principal, consultar página de suministros o consultar página del perfil del usuario (Figura 28 - Menú inferior).

Figura 27 - Menú superior

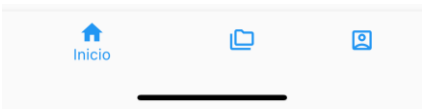


Figura 28 - Menú inferior

[7] PVPC: Precio Voluntario para el Pequeño Consumidor

Perfil del usuario

En la pantalla del usuario se mostrará la información relativa al usuario, la cual en esta versión es limitada, pero podría escalarse para implementar futuras funcionalidades y un botón que permite cerrar la sesión del usuario.

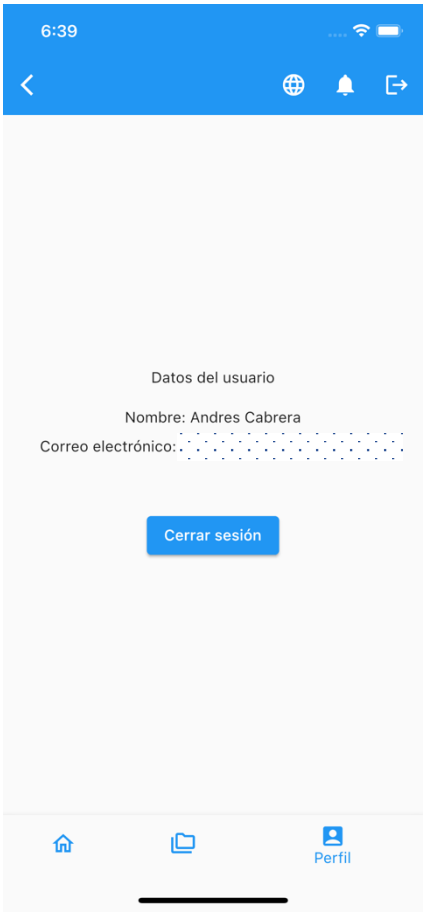


Figura 29 - Perfil del usuario

Visualización de suministros

En este apartado podremos visualizar los suministros que se hayan sincronizado con nuestra cuenta a través de Datadis, por lo tanto, en caso de estar vacío habrá algún problema con la plataforma, o que la contraseña introducida en la plataforma de Datadis no sea correcta.

La forma de visualizar los datos consiste en una lista cuyos componentes vienen formados por celdas en las que se encuentra la información mínima del punto de suministro acompañada de dos botones. El primero de ellos para tener un acceso a la visualización del contrato con más detalle (Consultar contrato) y por otro lado un acceso para poder visualizar los consumos de dicho suministro en una fecha determinada (Consultar consumos).

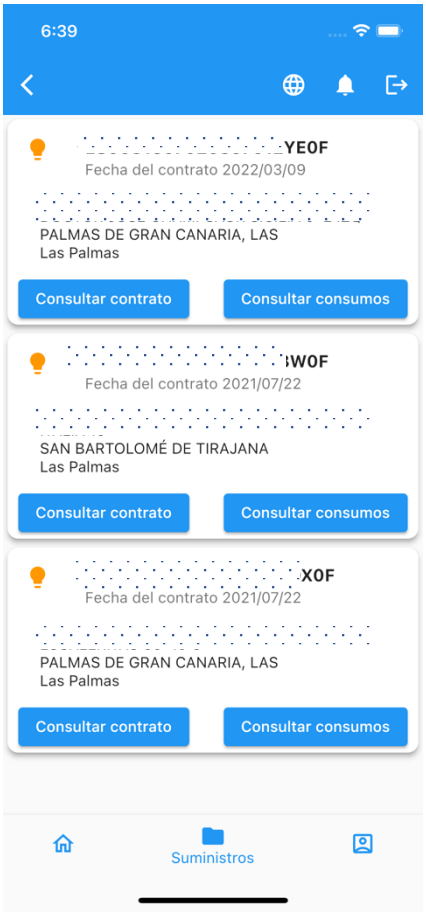


Figura 30 - Mis suministros

Visualización de contrato

En esta pantalla se tendrá, como se comentó en el epígrafe anterior, una visualización más extensa y detallada del consumo seleccionado.

Para ello se ha utilizado un formato en el que los elementos a mostrar se distribuyen de forma lineal por pantalla, para que todo sea visualmente alcanzable al usuario sin tener que realizar ningún clic.

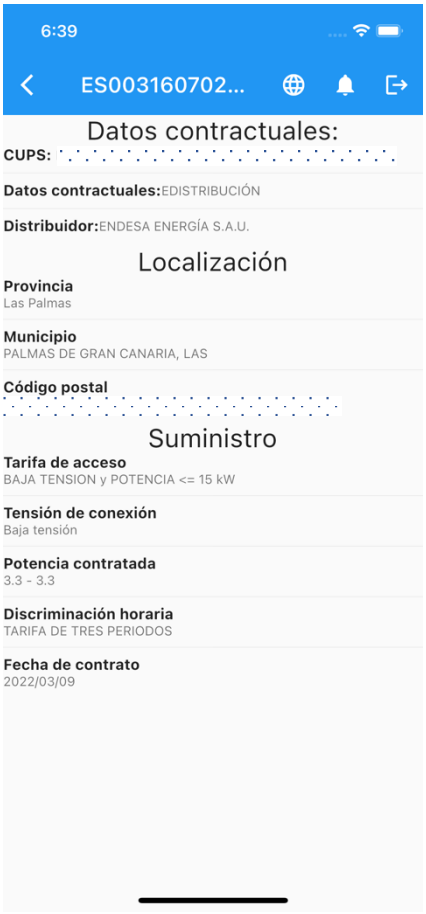


Figura 31 - Contrato de un suministro

Visualización de consumos

En esta pantalla se muestra un calendario en el cual podremos decidir el tipo de consumo a consultar, del cual disponemos de dos opciones, siempre teniendo en cuenta que la fecha deberá ser anterior al día actual por razones obvias. Para alternar entre los modos, disponemos de un botón a la izquierda:

- **Modo día**: Modo por defecto, en el que seleccionaremos un día. Al seleccionar esta opción se mostrará el consumo para las 24 horas del día. (Figura 32)
- **Modo rango**: Habrá que cambiar el modo, tras ello podremos seleccionar una fecha de inicio y otra final. Al seleccionar está opción se mostrará el consumo resumido por día, el cual equivale, a la suma de los consumos durante dicho día. (Figura 33)

Para mostrar el calendario se ha utilizado el paquete `table_calendar`[8] el cual permite implementar un calendario y personalizarlo a nuestro gusto.

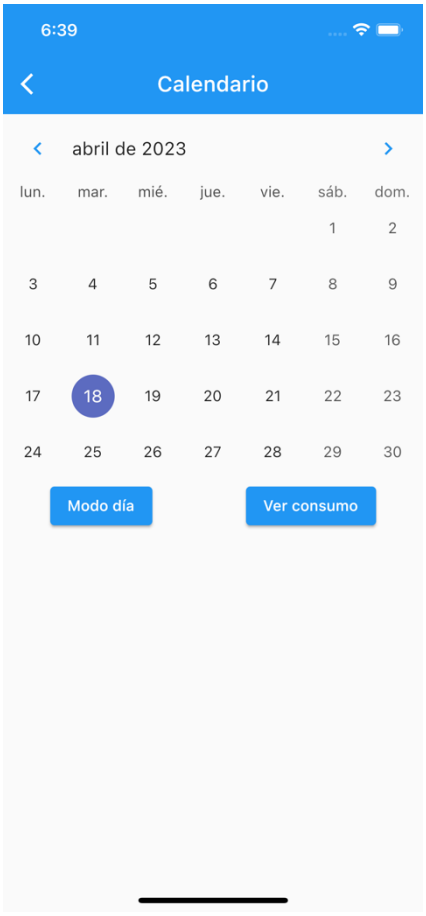


Figura 32 - Consultar suministro modo día

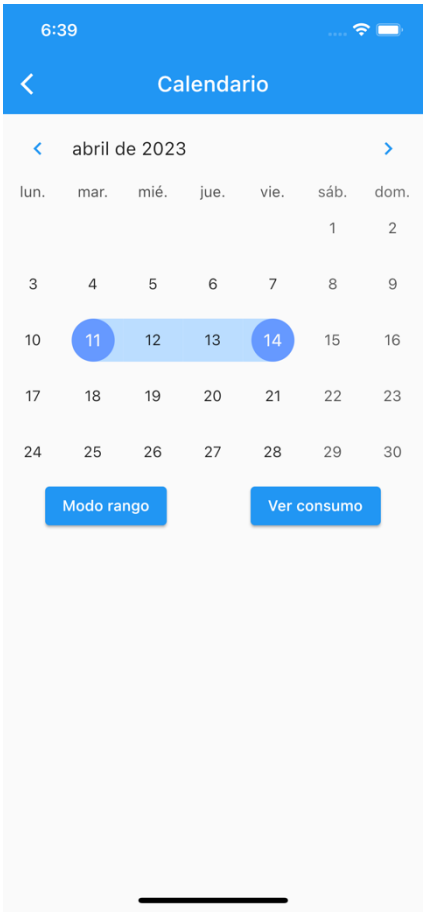


Figura 33 - Consultar suministro modo rango

Tras seleccionar el día y hacer clic en consultar consumo se generará un gráfico de barras dependiendo del modo, en el cual se muestra el consumo por puntos. Además, en caso de ser en modo de día, se permitirá comparar el resultado con los demás suministros alojados en la base de datos.

[8] www.pub.dev/packages/table_calendar

Para ello, se muestra un menú desplegable inferior con dos opciones:

- **Provincia**: Obligatorio cumplimentación, para poder seleccionar con el área a comparar.
- **Municipio**: Tras seleccionar el campo anterior, se actualizará los municipios disponibles con suministros para dicha selección, dicho campo no es obligatorio, pero se dispone de él en caso de querer comparar un área más específica.

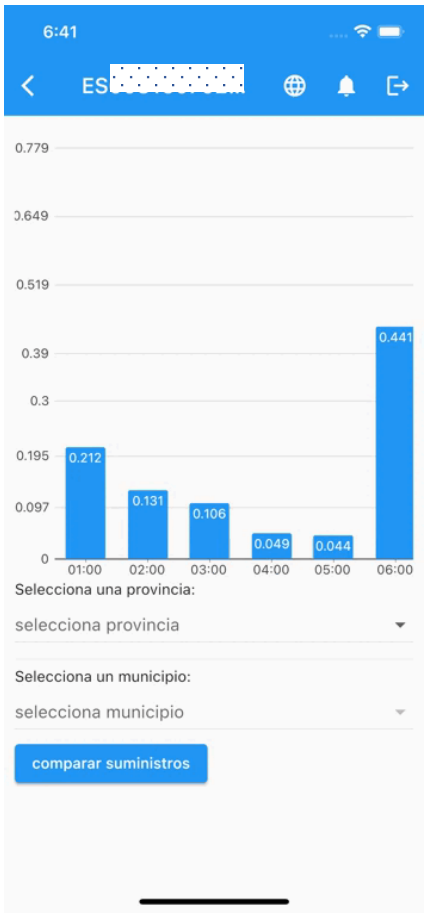


Figura 34 - Consumo modo día

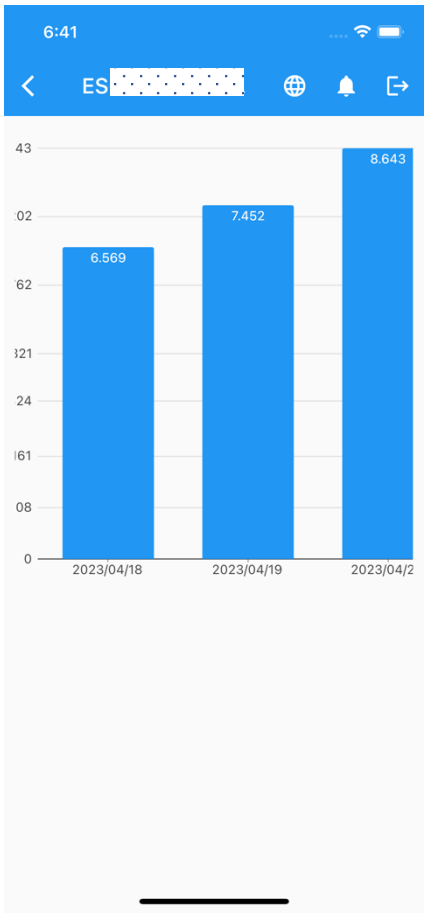


Figura 35 - Consumo modo rango

Comparación de consumos

Tras consultar el consumo para un día concreto podremos además comparar nuestro consumo para esas horas, con los demás suministros del área seleccionada.

El resultado será una lista, conformado por 3 columnas, con la siguiente disposición:

- **Hora**: Punto horario registrado del consumo.
- **Consumo**: Consumo propio obtenido de la pantalla anterior.
- **Comparado**: Consumo promedio del filtro aplicado en la pantalla anterior.

Aquellas horas que muestren un consumo mayor al nuestro de media estarán seleccionadas de color verde, mientras que aquellas cuyo consumo promedio es menor estarán de color rojo, como se indica en la leyenda de la misma pantalla.

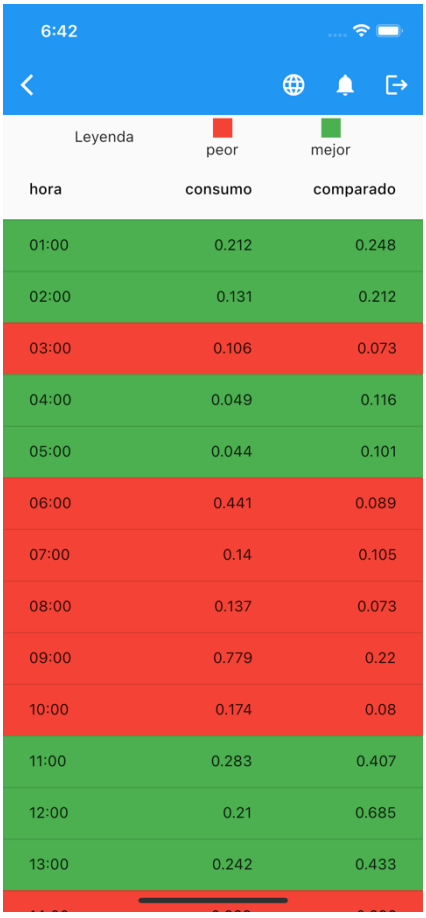


Figura 36 - Comparación de suministro

6.4 Elección del nombre de la aplicación

Durante el desarrollo del trabajo se fue barajando la propuesta de darle un nombre a este producto con el fin de darle una esencia y personalidad de un producto real y no dejarlo en un trabajo más. En cuanto a la elección del nombre, en un primer momento se tomó como referencia para su nombre a la conocida aplicación de Spotify, si bien ambas no tienen ningún nexo en común, su nombre daría cabida a ser fácil de recordar para los futuros usuarios. Tras un par de iteraciones y distintas variantes surgió el nombre de "Enotify".

"Enotify" ha sido seleccionado cuidadosamente para reflejar la principal función y propósito de la aplicación. La combinación de las palabras "En" de la raíz de "Energía" y "notify" haciendo referencia a la palabra anglosajona de "notificar" logra el objetivo de transmitir la idea de estar informado y conectado de manera instantánea. El nombre se ha elegido con el objetivo de captar la atención del usuario y transmitir la naturaleza ágil y eficiente de la aplicación. Además, es fácil de recordar y pronunciar, lo que contribuye a su accesibilidad y reconocimiento. "Enotify" encapsula la esencia de la aplicación, ofreciendo una experiencia notificativa y oportuna, manteniendo a los usuarios actualizados y conectados con la información relevante en tiempo real.

6.5 Diseño del logotipo de la aplicación

Durante el proceso de desarrollo del logo para la aplicación "Enotify", se ha puesto especial énfasis en capturar la esencia y el propósito de la aplicación de manera impactante y distintiva.

Con ello se propuso mantener el estándar actual de diseño de logotipos de aplicaciones, con formas redondeadas, nombre sencillo y algún icono que representara gráficamente lo que se quería transmitir. Tras iterar con distintas ideas y probar a integrar algún icono dentro del nombre, se llegó a la siguiente idea:

Producir a la sustitución de la letra "o" por el icono de una bombilla encendida, simbolizando la idea de notificaciones e ideas brillantes, manteniendo el resto de la palabra para que la identidad del nombre de la marca no perdiera fuerza. Dando como resultado preliminar al siguiente diseño resultante:

Figura 37 - Primer diseño del logotipo de Enotify

Pero los colores elegidos y su tonalidad no terminaban de encajar con el diseño ya implementado dentro de la interfaz de la aplicación. En un segundo iterado del mismo se intentó simplificar los colores, logrando mantener una tonalidad más seria y que a la vez no resultara tan llamativa, pero lograra transmitir el mismo mensaje:

Figura 38 - Mejora sobre el logotipo de Enotify

El resultado final tras una tercera iteración fue mantener una tonalidad negra en sus letras, y dándole todo el énfasis al propio icono introducido, modificando además la fuente utilizada, por una más rectangular sin bordes redondeados. Logrando incluso mejorar y transmitir la energía y atención deseada con el producto.

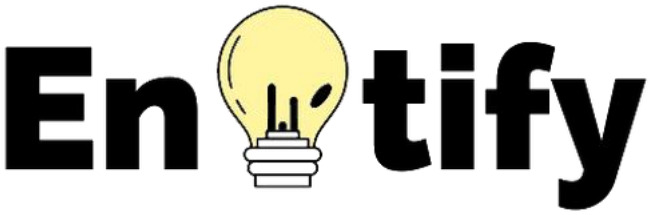


Figura 39 - Logotipo final de la aplicación Enotify

El diseño del logo fue cuidadosamente elaborado, combinando colores vibrantes y modernos que transmiten energía y atención, pero manteniendo el actual minimalismo que existe dentro del mercado. El resultado es un logo único y memorable que refleja la funcionalidad y el espíritu innovador de la aplicación "Enotify" y aporta un valor extra al producto.

Finalmente se generó los distintos formatos necesarios, tanto para su versión de iOS como Android con variación de tamaños, para en caso futuro ser publicada sus “Marketplace” correspondientes.

6.6 Integración con la API de Datadis

En este apartado se enfocará en el proceso clave y fundamental del funcionamiento y utilización de la API de **Datadis**, la cual nos permite acceder a la información de consumo almacenada en las bases de datos de las distintas distribuidoras eléctricas, utilizando una única API. Podemos ver datos de suministros propios o de terceros que nos han autorizado o datos agregados por zonas, sectores o potencias máximas.

Esta API ha sido creada utilizando las definiciones propias de “RestfulAPI”, por lo que para ello es necesario utilizar herramientas diseñadas para realizar consultas HTTP o algún software que integre dichas llamadas como es el caso.

Para utilizar la API de **Datadis** es necesario estar registrado en la plataforma ya que para consumir cualquier recurso se debe disponer de un ‘*token’*, obtenido a través de la llamada expuesta en la sección de Autenticación, el cual se insertará en la cabecera de la llamada a través del encabezado Autorización seguido de ‘*Bearer <token>’*.

Cabe destacar que dicha API está diseñada para solo realizar consultas para obtener datos (**GET**), en ningún caso se podrá realizar un método de inserción (**POST**) o modificación (**PUT**).

1.1.1.1 Respuestas y errores

Datadis utiliza los códigos de respuesta http habituales en este tipo de recursos para saber si se ha completado satisfactoriamente la solicitud requerida.

Dentro de los mismos existen 5 tipos de respuesta diferentes:

- 2xx Respuestas satisfactorias
    - **200 OK**
- 4xx Errores por parte del cliente
    - **400 Bad request**
    - **401 Unauthorized**
    - **403 Forbidden**
    - **404 Not Found**
    - **429 Too Many Request**: Esta tendrá especialmente hincapié, pues para consultar los consumos solo se podrá realizar una misma llamada en un periodo de 24 horas, es decir si consultamos el consumo para un suministro en febrero de 2023, no podremos volver a consultarlo hasta que haya transcurrido 24 horas desde la solicitud anterior.
- 5xx Errores por parte del servidor
    - **500 Internal Error Server**

Para la realización de estas llamadas y comprobar su correcto funcionamiento se utilizó en una primera etapa la herramienta Postman.

1.1.1.2 Autentificación POST

Obtiene el '*token*' de autenticación para el área privada.

Parámetros:

- username *String* **required**: NIF del usuario dado de alta en Datadis.
- password *String* **required** : Contraseña de acceso a Datadis del usuario.

https://datadis.es/nikola-auth/tokens/login

1.1.1.3 /get-supplies GET

Buscar todos los suministros del usuario.

Parámetros:

- authorizedNif *String*: Si queremos buscar suministros de personas que hemos autorizado, podemos buscarlo con el NIF de las personas autorizadas.
- distributorCode *String*: Código del distribuidor.

https://datadis.es/api-private/api/get-supplies

<pre>[
  {
  "address" : "C/ MORIONES, 23 , 1ºG 03182-TORREVIEJA - ALICANTE",
  "cups" : "ES002100xxxxxxxxxxJL",
  "postalCode" : "3182",
  "province" : "Alicante",
  "municipality" : "TORREVIEJA",
  "distributor" : "I-DE REDES ELÉCTRICAS INTELIGENTES, S.A.U.",
  "validDateFrom" : "2021/11/29",
  "validDateTo" : "2023/11/29",
  "pointType" : 5,
  "distributorCode" : "8"
  }
]</pre>

Tabla 12 - Ejemplo respuesta /get-supplies

1.1.1.4 /get-contract-detail GET

Buscar el detalle de un contrato.

Parámetros:

- cups *String* **required**: cups del contrato a consultar.
- authorizedNif *String*: Si queremos buscar suministros de personas que hemos autorizado, podemos buscarlo con el NIF de las personas autorizadas.
- distributorCode *String* **required**: Código del distribuidor.

https://datadis.es/api-private/api/get-contract-detail

<pre>[
  {
  "cups" : "ES002100xxxxxxxxxxJL",
  "distributor" : "I-DE REDES ELÉCTRICAS INTELIGENTES, S.A.U.",
  "marketer" : "TOTALENERGIES MERCADO ESPAÑA, S.A.",
  "tension" : "Baja tensión",
  "accessFare" : "BAJA TENSION y POTENCIA <= 15 kW",
  "province" : "Madrid",
  "municipality" : "MADRID",
  "postalCode" : "30740",
  "contractedPowerkW" : [3.45, 3.45],
  "timeDiscrimination" : "",
  "modePowerControl" : "ICP",
  "startDate" : "2022/09/13",
  "endDate" : ""
  }
]</pre>

Tabla 13 - Ejemplo respuesta /get-contract-detail

1.1.1.5 /get-consumption-data GET

Buscar los datos de consumo.

Parámetros:

- cups *String* **required**: cups del contrato a consultar.
- authorizedNif *String*: si queremos buscar suministros de personas que hemos autorizado, podemos buscarlo con el NIF de las personas autorizadas.
- distributorCode *String* **required**: código del distribuidor.
- startDate *String* (date-time) **required**: fecha de inicio entre los datos de búsqueda. Formato: AAAA/MM.
- endDate *String* (date-time) **required**: fecha de final entre los datos de búsqueda. Formato: AAAA/MM.
- measurementType *String* **required**: valor que por defecto siempre será 0 en nuestro caso, para obtener los consumos por horas.
- pointType *String* **required**: código de tipo de punto, obtenido en la solicitud anterior.

https://datadis.es/api-private/api/get-consumption-data

<pre>[
  {
  "cups" : "ES002100xxxxxxxxxxJL",
  "date" : "2022/07/01",
  "time" : "01:00",
  "consumptionKWh" : 4879.0,
  "obtainMethod" : "Real"
  }
]</pre>

Tabla 14 - Ejemplo respuesta /get-consumption-data

Finalmente destacar que la API de **Datadis** actualmente, sigue añadiendo funcionalidades nuevas a su API, y puede que la versión actual en este documento haya sufrido modificaciones en su uso, por lo que se anima al lector a consultar la propia documentación de la API si desea hacer uso de la misma. En el Anexo IV - Código fuente API se puede consultar como se desarrolló el mismo y las llamadas descritas a lo largo de esta sección.

6.7 Pruebas

Para el testeo en esta aplicación se ha hecho uso frecuente de las excepciones, eventos que se generan durante el proceso de ejecución al ocurrir ciertos eventos que impiden el flujo normal de la aplicación, además de la validación de campos en los formularios.

Concretamente se han usado en los métodos que realizan llamadas a la API, o creando excepciones propias. En el caso de producirse estas deben ser tratadas de manera especial, un ejemplo de ello se produce al intentar registrar a un usuario.

En el proceso de registro el usuario introducirá un NIF y una contraseña para el acceso a **Datadis**, pues si al ingresar un NIF incorrecto, o que dicho NIF no esté registrado en la base de datos de **Datadis** se lanzará una excepción evitando que dicho usuario pueda continuar con su registro e informando al mismo de dicho error para que pueda subsanar en caso deseado.

A continuación, se muestra el código que lanza una excepción al no poder realizar la autenticación a través de **Datadis**.

```
final response = await http.post(uri.replace(queryParameters: queryParams),
      headers: headers);
  return response.statusCode == 200
      ? response.body
      : throw Exception('Failed to log in');
```

Por otro lado, en la parte donde se realiza el registro del usuario, se procederá a envolver el código en una estructura “try-catch”, para en caso de error no proceder a la inserción del usuario y por otro lado mostrarle un mensaje de error.

```
Future<String?> _signupUser(SignupData data) async{
  Database database = await DB.openDB();
  var tempUser = await database.rawQuery("SELECT * FROM users WHERE email = '${data.name}'
AND password = '${data.password}' OR nif = '${data.additionalSignupData!['nif']}';");
  User user = User.empty();
  try{
    if(tempUser.isEmpty){
    String encryptPassword = encryptMyData(data.password!);
    String encryptDatadisPassword = encryptMyData(data.additionalSignupData!['datadisPassword']!);
    user = User(email: data.name!, password: encryptPassword, name:
data.additionalSignupData!['name']!, surname: data.additionalSignupData!['surname']!, nif:
data.additionalSignupData!['nif']!, datadisPassword: encryptDatadisPassword);

    String token = await API.postLogin(data.additionalSignupData!['nif']!,
data.additionalSignupData!['datadisPassword']!);
    final prefs = await SharedPreferences.getInstance();
    prefs.setString("datadisToken", token);
    prefs.setString("email", user.email!);
  }else{
    return Future.delayed(loginTime).then((_) {
      return 'Error code: DNI ${data.additionalSignupData!["nif"]!} ya registrado en el sistema';
    });
  }
  }catch(exception){
```

```
    user = User.empty();
    return Future.delayed(loginTime).then((_) {
      return 'Error code: usuario no registrado en Datadis';
    });
  }finally{
    if(user.email != null) database.insert("users", user.toMap());
  }
}
```

6.8 Despliegue

En cuanto a la hora del despliegue de este proyecto, habría que tener en cuenta **3 fases** en el mismo, las cuales se detallaran a continuación.

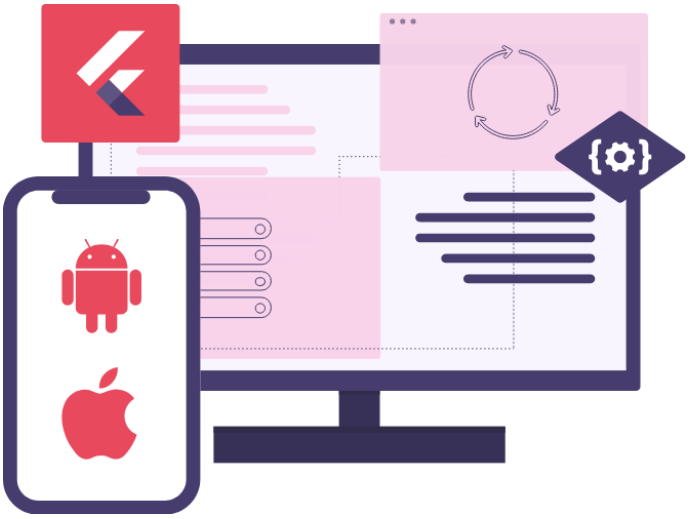


Figura 40 - Despligue aplicación. Fuente: Startech Up

Elección del Servidor

Una vez la aplicación se encuentre finalmente desarrollada y lista para despliegue, el primer paso será seleccionar un servidor donde alojarla, para ello debido al presupuesto limitado y ya que deseamos primero probar la viabilidad de este proyecto se decide por desplegar la misma, a través del servidor de **Firebase**. Está plataforma brinda almacenamiento y alojamiento gratuito para aplicaciones móviles, además en el caso de **Firebase** la posibilidad de vincular con otras herramientas de Google, lo que nos permite subir y mantener nuestra aplicación en línea sin incurrir en costos iniciales significativos.

Una vez ya hemos seleccionado nuestro servidor donde alojarla, deberemos crear una cuenta en la plataforma elegida y seguir los pasos proporcionados para subir y configurar nuestra aplicación. Esto incluye la carga de archivos, configuración de variables de entorno, y establecimiento de enlaces con los servicios externos y seguir los pasos proporcionados por el proveedor del servidor para asegurar su correcto despliegue.

Figura 41 - Logo Firebase. Fuente: Wikipedia

Marketplace

Una vez que nuestra aplicación está en funcionamiento en un servidor gratuito, es hora de considerar su distribución a un público más amplio. Los "marketplaces", como la App Store o Google Play Store, se han convertido en los portales más populares para que los usuarios descarguen aplicaciones móviles. Estas plataformas brindan una exposición masiva y nos permiten llegar a millones de usuarios en todo el mundo. Para publicar nuestra aplicación en un "Marketplace", deberemos cumplir con los requisitos y directrices específicos establecidos por cada plataforma, además de pagar las tasas pertinentes de suscripción como es en el caso de la App Store. Esto incluye la preparación de una descripción convincente, capturas de pantalla atractivas y el cumplimiento de los estándares de calidad y seguridad establecidos por la plataforma.

Existen otras alternativas más nuevas como pueden ser el servicio ofrecido por Amazon, pero sí que es verdad que su posicionamiento no es igual de grande que el de los anteriores, pero sí cabe destacar que en principio dicho servicio sería gratuito.

Figura 42 - Logo App Store. Fuente: Wikipedia

Figura 43 - Logo Play Store. Fuente: Wikipedia

Mantenimiento y actualizaciones

Por último, e igual más importante, una vez que hemos desplegado nuestra aplicación en un servidor gratuito y la hemos lanzado en un "Marketplace", es importante recordar que el trabajo no termina ahí. El mantenimiento continuo y las actualizaciones son esenciales para garantizar un rendimiento óptimo de la aplicación y una experiencia de usuario satisfactoria. Esto implica supervisar el servidor, solucionar problemas técnicos, realizar mejoras de seguridad y lanzar actualizaciones periódicas que incorporen nuevas funcionalidades o correcciones de errores. Además, es fundamental seguir actualizando la integración con la API a medida que esta vaya evolucionando.

Capítulo 7 Conclusiones y trabajos futuros

Introduciendo esta innovadora plataforma móvil, tendrás acceso a la consulta de suministros eléctricos de manera rápida y sencilla, sin importar el proveedor de servicios. Está aplicación móvil, compatible con cualquier dispositivo, ha sido desarrollada para permitir a los usuarios realizar todas las tareas mencionadas en las secciones anteriores.

Pero eso no es todo. Este proyecto ha sido una oportunidad de crecimiento personal, adquiriendo conocimientos en el desarrollo móvil con un lenguaje de programación de vanguardia, con una gran demanda en el mercado. Además, ha logrado fortalecer mis habilidades adquiridas a lo largo de la carrera.

En resumen, los objetivos planteados inicialmente han sido cumplidos con éxito, incluso se han agregado nuevas funcionalidades durante el desarrollo para brindar un mayor valor al producto. Y, para futuros proyectos, podríamos seguir mejorando y añadiendo características que brinden aún más utilidad y valor. Incluso considerar la posibilidad de ofrecer esta aplicación de forma gratuita para el público en general una vez se produjera su lanzamiento, con el fin de adquirir una gran fuente de datos.

Entre las emocionantes características futuras que se pudieran implementar se encuentran:

1. **Inicio de sesión a través de redes sociales**

Esto daría la posibilidad a los usuarios de que pudieran acceder de manera más sencilla al registro en la plataforma mediante las cuentas de Google y Facebook. Incrementando de igual manera la seguridad en la conexión, ya que dichos sistemas siguen un estándar de seguridad más fuerte.

2. **Cumplir con la normativa del Reglamento Europeo de Protección de Datos (RGPD) o Ley Orgánica de Protección de Datos (LOPD)**

En este proyecto se ha intentado mantener de manera encriptada la información sensible de los datos que se introducen en la base de datos, pero, en cualquier caso, no cumple con la normativa europea actual. Por ello sería interesante asegurar el cumplimiento de la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales [9]. Con esta ley se persigue el fin de mejorar la privacidad y seguridad de los datos de los usuarios, la creación de una política de privacidad y de tratamiento de los datos del usuario. Para ello deberá haber un encargado personalmente jurídico encargado de los mismos.

3. **Hacer uso de un servidor para alojar la información**

Con este fin se persigue que los datos están centralizados en un lugar externo al dispositivo del usuario, asegurando de dicha manera la información de los mismos, y evitando en caso de pérdida de los datos en un dispositivo todos los datos que se hayan almacenado en el mismo. Aplicando con ello la arquitectura mencionada en este documento de cliente-servidor.

4. **Implementar notificaciones “push” en vez de notificaciones locales**

En esta fase de desarrollo se llegó a implementar una serie de notificaciones las cuales, debido a las limitaciones del mismo, solo se producen de forma local al estar manipulando la aplicación. Con esta ampliación se conseguiría, independientemente de si el usuario está usando o no la

aplicación programar notificaciones a ciertas horas del día informando en este caso del precio máximo del PVPC del día, para aportar información relevante al usuario sin necesidad del mismo de tener que consultarlo cada día.

5. Permitir dar de baja al usuario

En caso de que el usuario dejará de estar interesado en estar compartiendo sus datos en la aplicación, se deberá permitir dar de baja y borrar todos los datos asociados al usuario con el fin de cumplir con la normativa de protección de datos.

6. Permitir recuperar las contraseñas de los usuarios

Debido a la gestión local de los datos, en la actualidad no es posible solicitar a través de un email, un cambio de contraseña de la cuenta sería por ello interesante usar algún método que permitiera a los usuarios en caso de olvidar su contraseña, volver a solicitar una al sistema, para poder acceder a su cuenta en caso de haberla olvidado.

Algunas de estas funcionalidades como la 1, 3, 4, 7 podrían ser gestionadas por ejemplo a través de Firebase, pero como se comentó a lo largo de este proyecto se decidió descartar por dicho sistema, pues hacía más lento la carga de la aplicación y para el estado del desarrollo de este proyecto, no eran necesaria llegar a tal punto de implementación, pero podría ser una de las posibles soluciones exploradas. Además, con esta opción al disponer de una base de datos no relacional, se podría además alojar los consumos de distintas formas, y no únicamente como datos de una base de datos relacional.

Anexo I Competencias Específicas

Las competencias generales del grado y de la mención de tecnologías de la información que tienen relación con el desarrollo de este trabajo son las siguientes. [10]

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."

Esto se ha logrado aplicando un enfoque exhaustivo en la creación de una aplicación móvil sólida, segura y eficiente, seleccionando cuidadosamente los lenguajes de programación y paradigmas más apropiados.

CII016: "Conocimiento y aplicación de los principios, metodologías y ciclos de vida de la ingeniería de software."

Durante el proceso de desarrollo, he aplicado los principios, metodologías y ciclos de vida de la ingeniería de software para garantizar la calidad y coherencia en todas las etapas del proyecto.

CII017: "Capacidad para diseñar y evaluar interfaces persona computador que garanticen la accesibilidad y usabilidad a los sistemas, servicios y aplicaciones informáticas."

He diseñado y evaluado cuidadosamente las interfaces de usuario de la aplicación, asegurando que sean fáciles de utilizar.

En cuanto a las específicas que son propia de la rama de Ingeniería del Software:

IS01: "Capacidad para desarrollar, mantener y evaluar servicios y sistemas software que satisfagan todos los requisitos del usuario y se comporten de forma fiable y eficiente, sean asequibles de desarrollar y mantener y cumplan normas de calidad, aplicando las teorías, principios, métodos y prácticas de la ingeniería del software."

A lo largo del proyecto, he aplicado las teorías, principios y prácticas de la ingeniería del software para desarrollar, mantener y evaluar la aplicación, asegurando que cumpla con todos los requisitos del usuario y los estándares de calidad.

IS02: "Capacidad para valorar las necesidades del cliente y especificar los requisitos software para satisfacer estas necesidades, reconciliando objetivos en conflicto mediante la búsqueda de compromisos aceptables dentro de las limitaciones derivadas del coste, del tiempo, de la existencia de sistemas ya desarrollados y de las propias organizaciones."

He evaluado cuidadosamente las necesidades del "product owner" a través de reuniones y he especificado los requisitos de software correspondientes, teniendo en cuenta los diferentes objetivos y limitaciones que se presentaron a lo largo del proyecto.

IS03: "Capacidad de dar solución a problemas de integración en función de las estrategias, estándares y tecnologías disponibles."

Durante el desarrollo de la aplicación, aborde los desafíos de integración utilizando estrategias, estándares y tecnologías disponibles, asegurando una integración fluida.

IS05: ”Capacidad de identificar, evaluar y gestionar los riesgos potenciales asociados que pudieran presentarse.”

Esto fue posible gracias a evaluar cómo se exponían los datos sensibles como las contraseñas y su solución introduciendo un método de encriptación de las mismas.

Además, se ha adquirido la capacidad específica de la mención de Tecnologías de la Información:

TI06: ”Capacidad de concebir sistemas, aplicaciones y servicios basados en tecnologías de red, incluyendo Internet, web, comercio electrónico, multimedia, servicios interactivos y computación móvil.”

He aplicado mi conocimiento en tecnologías de red, como Internet, web y servicios interactivos para desarrollar una aplicación móvil que haga uso de estos conocimientos.

A modo de conclusión final, a través de este proyecto, he adquirido y aplicado un amplio conjunto de competencias en el campo de la ingeniería del software y las tecnologías de la información. Estas competencias me han permitido desarrollar una aplicación móvil robusta, segura y eficiente, que cumple con los requisitos del usuario, garantiza la usabilidad y accesibilidad, y se adhiere a los estándares de calidad y prácticas de la industria.

Anexo II Manual de Usuario

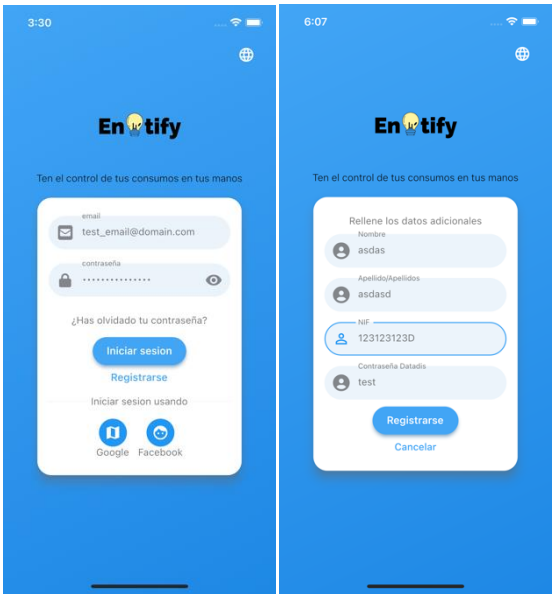
Remarcar que, en todas las capturas adjuntas, por temas de privacidad se han ocultado ciertos campos de información sensible, como identificadores de cups y direcciones de los suministros.

I. Requisitos previos

Los requisitos necesarios para poder acceder a la plataforma serán:

- Un dispositivo donde simular la aplicación o un dispositivo físico.
- Registro previo a través de la plataforma de Datadis.
- Acceso a internet para disponer de conexión a las APIs.

II. Iniciar sesión y registro



Tras el registro en la plataforma de Datadis, se debe abrir la aplicación y la primera pantalla que se mostrará será la de autenticación del usuario. En caso de no estar registrado pulsando el botón de Registro y rellenando el formulario superior previamente podremos registrarnos en el sistema rellenando el siguiente formulario, con nuestros datos y los datos de acceso a Datadis.

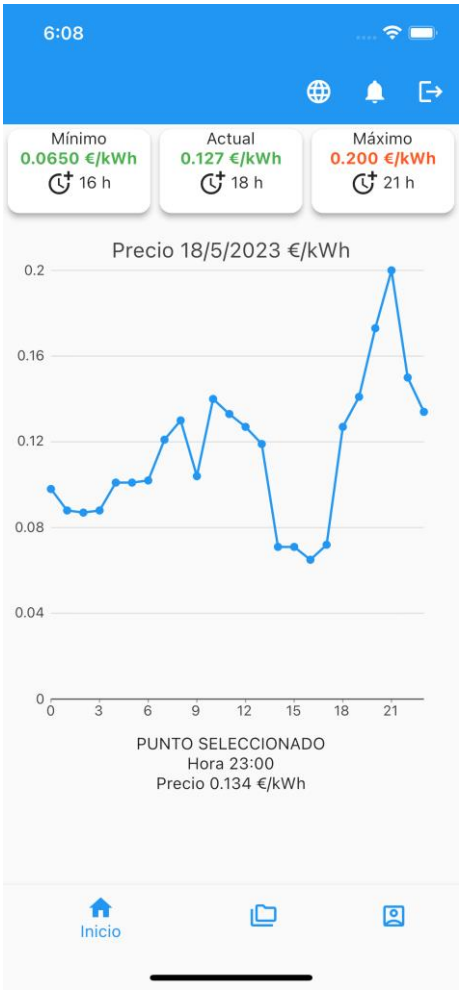
Tras completar los datos, se pulsará en el botón de *Registrarse*. Si todo ha ido bien, será redirigido a la página principal de la aplicación, en caso contrario, se mostrará un mensaje de error.

Para iniciar sesión, deberá pulsar en el botón de *Iniciar Sesión* que se encuentra en la pantalla rellenando previamente los datos de acceso. Al igual que en el registro, si el proceso se completa exitosamente, se redirigirá a la página inicial o se mostrará un mensaje de error en la parte inferior.

III. Visualizar PVPC del día actual

Una vez se haya cargado la página principal, en esta podemos desde cambiar el idioma de la aplicación, consultar el precio de mercado regulado en el día de hoy en cualquiera de sus tramos y además visualizar directamente los puntos más críticos, es decir, aquellos en los que el precio es mayor y menor en el día. Además, haciendo clic en alguno de los puntos de la gráfica se mostrará debajo el precio para dicha hora en €/kWh.

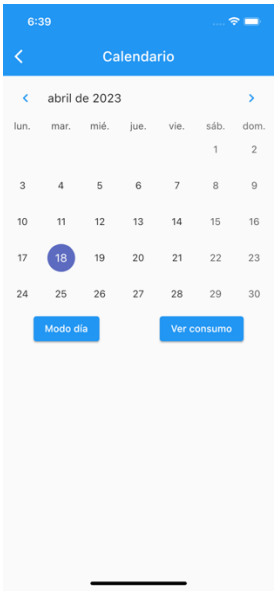
Por otro lado, en la barra inferior, tenemos el menú de navegación que nos permite navegar entre las distintas vistas principales: Inicio, Suministros y Perfil.

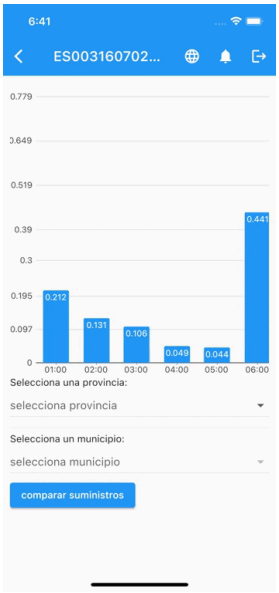


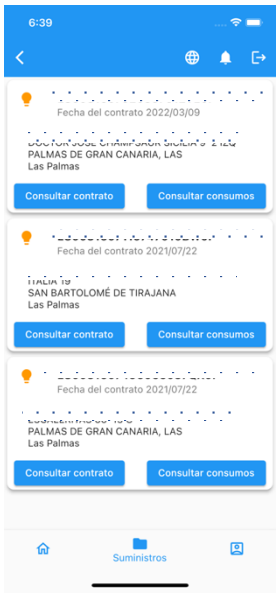
IV. Consultar suministros

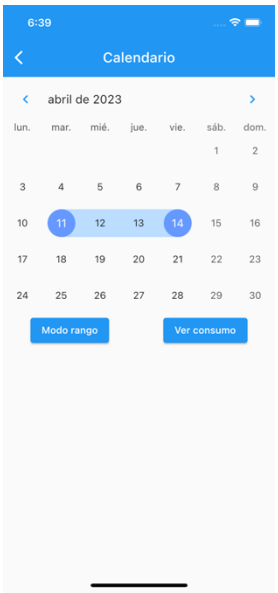
Para consultar el consumo de un suministro navegaremos a la página de suministros, donde se desplegará una lista de los suministros asignados a nuestro usuario, si hacemos clic en el botón pertinente se nos mostrará una vista con un calendario donde podremos seleccionar el día o días a consultar.

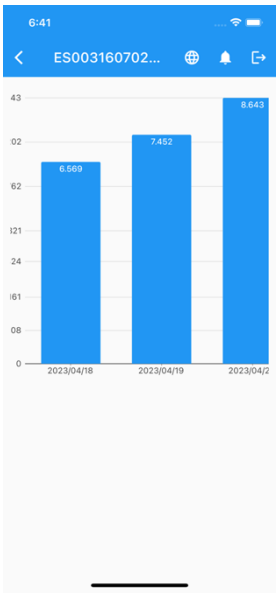
Una vez hagamos esto en función del tipo de rango seleccionado se nos mostrará una u otra información. En caso de la primera se mostrarán 24 barras, una para cada hora del día con sus consumos en kWh respectivamente. En el caso de haber seleccionado más de un día, se mostrará n-barras verticales, una para cada día representando el consumo total de dicho día.



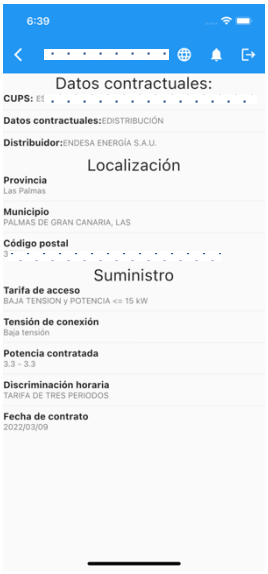








V. Consultar contrato de un suministro

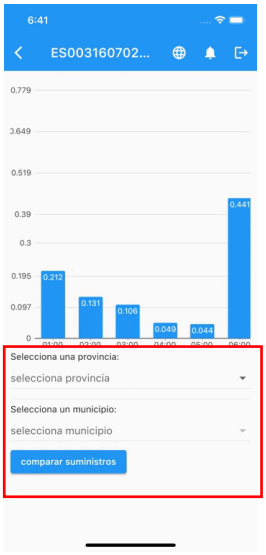


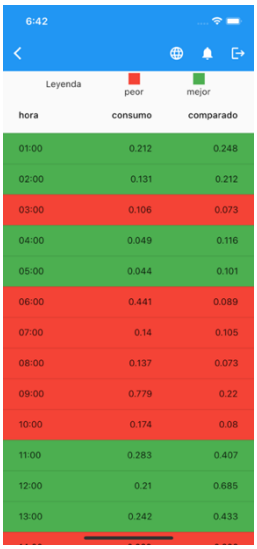
De igual manera para consultar el contrato de un suministro navegaremos a la página de suministros, donde se desplegará una lista de los suministros asignados a nuestro usuario, si hacemos clic en el botón pertinente se nos mostrará una vista con toda la información relativa obtenida acerca del suministro.

VI. Comparación de suministro

Finalmente, cuando consultamos el consumo de un suministro para un día, tendremos la opción de comparar los resultados obtenidos con los del resto de suministros de los que se disponga datos a través de la base de datos.

Para ello disponemos en la parte inferior de un selector de provincia el cual será de obligatorio cumplimiento a la hora de realizar la consulta, y tras seleccionar el mismo, se mostrará en el siguiente seleccionable de carácter voluntario un desplegable de municipios para los cuales también se dispone de datos a comparar.





VII. Cerrar sesión

Finalmente se permitiría cerrar la sesión del usuario en caso deseado a través de dos lugares, o directamente en la barra superior de navegación, o por su parte a través de la página del perfil, donde habrá igualmente un botón de *Cerrar Sesión.*

Anexo III Historias de usuario

A continuación, se mostrarán las historias de usuario que no fueron expuestas en la memoria principal para no alargar la lectura del documento de manera innecesaria, estas historias de usuario complementan el total del producto desarrollado:

ID	HU04
TITULO	Visualizar lista de suministros
ROL	Usuario registrado
QUIERO	Visualizar en la pantalla asignada a los suministros una lista de los mismos.
PARA	Poder saber que suministros tengo a disposición
CRITERIO DE ACEPTACION	- Debe estar autentificado en la aplicación. - Se mostrará en el menú de navegación un acceso a la página de visualización de suministros. - Se mostrará un listado de los suministros que el usuario tiene asignado.

Tabla 15 - HU04 Visualizar suministros

ID	HU08
TITULO	Visualizar contrato de un suministro
ROL	Usuario registrado
QUIERO	Visualizar la información relativa al contrato de un suministro
PARA	Poder consultar la información relativa del mismo.
CRITERIO DE ACEPTACION	- Debe estar autentificado en la aplicación. - Se mostrará para cada suministro un botón para consultar el contrato del suministro - Se abrirá una página nueva con la información del suministro.

Tabla 16 - HU08 Visualizar contrato para un suministro

ID	HU09
TITULO	Visualización Perfil
ROL	Usuario registrado
QUIERO	Visualizar la información de mi perfil
PARA	Poder consultar mi perfil o cerrar mi sesión
CRITERIO DE ACEPTACION	- Debe estar autentificado en la aplicación. - Se mostrará en el menú de navegación un acceso a la página del perfil - Se mostrará el email del usuario, el nombre. - Se mostrará un botón para Cerrar Sesión.

Tabla 17 - HU09 Visualizar perfil de usuario

ID	HU10
TITULO	Menú acceso rápido superior
ROL	Usuario registrado
QUIERO	Visualizar en la parte superior acceso rápido a distintas funcionalidades
PARA	Poder tomar acciones de forma rápida en cualquier punto.
CRITERIO DE ACEPTACION	- Debe estar autentificado en la aplicación. - Se mostrará en el menú de navegación superior una serie de botones de acceso rápido. - Se mostrará un botón para cambiar el idioma de la aplicación. - Se mostrará un botón para consultar las notificaciones. - Se mostrará un botón para cerrar la sesión.

Tabla 18 – HU10 Visualizar suministros

ID	HU11
TITULO	Aceptación de términos
ROL	Usuario no registrado
QUIERO	Asegurar que el usuario acepte los términos de esta app
PARA	Poder dar acceso a la lectura de sus datos a través de Datadis
CRITERIO DE ACEPTACION	- Debe ser un usuario sin registro en la aplicación. - Se mostrará a la hora de registrarse una casilla de marcación de aceptar los términos. - Si no se acepta esta casilla, no se permitirá continuar con el registro en el sistema.

Tabla 19 – HU11 Visualizar suministros

Anexo IV Código

En cuanto a las fuentes de código que se incluyen en este apartado, encontramos entre las mismas, aquellas que tienen una relevancia mayor o que han sido referenciadas a lo largo de los capítulos anteriores de la memoria. Entre estas se encuentran el funcionamiento de la base de datos (BBDD), métodos de acceso a las distintas API, como utilizar dichos métodos, ejemplo de creación de clases, creación de componentes reutilizables y la implementación y su utilización de las traducciones y su fichero de traducción.

I. Código fuente relacionado con operaciones de la base de datos

```
static Future<Database> openDB() async {
  var database = await openDatabase(
    join(await getDatabasesPath(), 'TFG.bd'),
    onCreate: (db, version) {
      db.execute("PRAGMA foreign_keys = ON");
      db.execute(
        "CREATE TABLE IF NOT EXISTS users(email TEXT PRIMARY KEY, password TEXT, name TEXT, surname TEXT, nif TEXT UNIQUE, datadisPassword TEXT);");
      db.execute(
        "CREATE TABLE IF NOT EXISTS supplies(cups TEXT PRIMARY KEY, address TEXT, postalCode TEXT, province TEXT, municipality TEXT, distributor TEXT ,validDateFrom TEXT, validDateTo TEXT, pointType INTEGER, distributorCode TEXT, userId TEXT, FOREIGN KEY(userId) REFERENCES users(email));");
      db.execute(
        "CREATE TABLE IF NOT EXISTS consumptions(id INTEGER PRIMARY KEY AUTOINCREMENT, date TEXT, time TEXT, consumptionKWh REAL, obtainMethod TEXT, supplyId TEXT, FOREIGN KEY(supplyId) REFERENCES supplies(cups));");
      db.execute(
        "CREATE TABLE IF NOT EXISTS contracts(cups TEXT PRIMARY KEY, distributor TEXT, marketer TEXT, tension TEXT, accessFare TEXT, province TEXT, municipality TEXT, postalCode TEXT, contractedPowerkWMin REAL, contractedPowerkWMax REAL, timeDiscrimination TEXT, startDate TEXT);");
    },
    version: 1,
    onOpen: (db) {
      db.execute("PRAGMA foreign_keys = ON");
      db.execute(
        "CREATE TABLE IF NOT EXISTS users(email TEXT PRIMARY KEY, password TEXT, name TEXT, surname TEXT, nif TEXT UNIQUE, datadisPassword TEXT);");
      db.execute(
        "CREATE TABLE IF NOT EXISTS supplies(cups TEXT PRIMARY KEY, address TEXT, postalCode TEXT, province TEXT, municipality TEXT, distributor TEXT ,validDateFrom TEXT, validDateTo TEXT, pointType INTEGER, distributorCode TEXT, userId TEXT, FOREIGN KEY(userId) REFERENCES users(email));");
      db.execute(
        "CREATE TABLE IF NOT EXISTS consumptions(id INTEGER PRIMARY KEY AUTOINCREMENT, date TEXT, time TEXT, consumptionKWh REAL, obtainMethod TEXT, supplyId TEXT, FOREIGN KEY(supplyId) REFERENCES supplies(cups));");
```

```
      db.execute(
        "CREATE TABLE IF NOT EXISTS contracts(cups TEXT PRIMARY KEY, distributor TEXT,
marketer TEXT, tension TEXT, accessFare TEXT, province TEXT, municipality TEXT, postalCode
TEXT, contractedPowerkWMin REAL, contractedPowerkWMax REAL, timeDiscrimination TEXT,
startDate TEXT);");
    },
  );
  return database;
 }
```

Tabla 20 - Código creación de BD

```
Database db = await DB.openDB();
```

Tabla 21 - Código apertura BD

```
Database database = await DB.openDB();
 final list = await database.query('contracts', where: 'cups = ?', whereArgs: [cups]);
 if(list.isEmpty){
  ContractDetail contractDetail = await API.getContractDetail(cups, distributorCode);
  var value = {
   'cups': contractDetail.cups,
   'distributor': contractDetail.distributor,
   'marketer': contractDetail.marketer,
   'tension': contractDetail.tension,
   'accessFare': contractDetail.accessFare,
   'province': contractDetail.province,
   'municipality': contractDetail.municipality,
   'postalCode': contractDetail.postalCode,
   'contractedPowerkWMin': contractDetail.contractedPowerkWMin,
   'contractedPowerkWMax': contractDetail.contractedPowerkWMax,
   'timeDiscrimination': contractDetail.timeDiscrimination,
   'startDate': contractDetail.startDate,
  };
  database.insert('contracts', value);
  return contractDetail;
 }else{
  ContractDetail contrato =  ContractDetail.fromJson(list.first);
  return contrato;
 }
```

Tabla 22 - Código operaciones sobre BD

```
final list = await database.query(
 'contracts',
 where: 'cups = ?',
 whereArgs: [cups]
);
```

Tabla 23 - Código query sobre BD

II. Código fuente API de REE y Datadis

```
class API {

 static Future<List<Price>> pricesToday() async{
  var today = DateTime.now();
```

```
    final response = await http.get(Uri.parse('https://apidatos.ree.es/es/datos/mercados/precios-
mercados-tiempo-real?start_date=${today.formatter2()}&end_date=${today.add(const Duration(days:
1)).formatter2()}&time_trunc=hour'));

    if(response.statusCode == 200){
      final json = jsonDecode(response.body);
      final jsonMap = json["included"][0]["attributes"]["values"];

      List<Price> prices = [];
      for (var element in jsonMap) {
        double precio = double.parse((element["value"] / 1000).toStringAsFixed(3));
        String datetime = element["datetime"];
        DateTime date = DateTime.parse(datetime.substring(0, datetime.indexOf("+")));
        if(date.day == today.day) {
          prices.add(Price(
            hour: date.toTwoDigits(date.hour),
            cheap: false,
            price: precio
        ));
        }
      }
      return prices;
    }else{
      throw Exception('Failed to fetch prices');
    }
  }

  static String apiBase = "https://datadis.es";
  static String apiBasePrivate = "https://datadis.es/api-private";

  /// Obtiene el token de autenticación para el área privada.
  /// @param username: NIF del usuario dado de alta en Datadis
  /// @param password: Contraseña de acceso a Datadis del usuario
  /// @return token de autentificación
  static Future<String> postLogin(String username, String password) async {
    final uri = Uri.parse('https://datadis.es/nikola-auth/tokens/login');
    final queryParams = {'username': username, 'password': password};
    final headers = {'Content-Type': 'application/json'};

    final response = await http.post(uri.replace(queryParameters: queryParams),
        headers: headers);
    return response.statusCode == 200
        ? response.body
        : throw Exception('Failed to log in');
  }

  ///Buscar todos los suministros
  static Future<List<Supply>> getSupplies() async {
    var prefs = await SharedPreferences.getInstance();
    String? token = prefs.getString('datadisToken');
    final url = Uri.parse('https://datadis.es/api-private/api/get-supplies');
```

```
  final headers = {
    'Content-Type': 'application/json',
    'Accept': 'application/json',
    'Authorization': 'Bearer $token'
  };

  final response = await http.get(url, headers: headers);

  if (response.statusCode == 200) {
    final String jsonResponse = utf8.decode(response.bodyBytes);
    final supplies =
        List<Map<String, dynamic>>.from(jsonDecode(jsonResponse));
    return supplies.map((json) => Supply.fromJson(json)).toList();
  } else {
    throw Exception('Failed to load supplies');
  }
}

///Obtener detalles de un suministro
static Future<ContractDetail> getContractDetail(
    String cups, String distributorCode) async {
  var prefs = await SharedPreferences.getInstance();
  String? token = prefs.getString('datadisToken');
  final uri =
      Uri.parse('https://datadis.es/api-private/api/get-contract-detail');
  final headers = {
    'Content-Type': 'application/json',
    'Accept': 'application/json',
    'Authorization': 'Bearer $token'
  };
  final queryParams = {'cups': cups, 'distributorCode': distributorCode};

  final response = await http.get(uri.replace(queryParameters: queryParams),
      headers: headers);

  if (response.statusCode == 200) {
    final jsonResponse = jsonDecode(utf8.decode(response.bodyBytes));
    ContractDetail contractDetail = ContractDetail.fromJson(jsonResponse[0]);
    return contractDetail;
  } else {
    throw Exception('Failed to load Contract Detail');
  }
}

///Obtener consumos de un suministro
static Future<List<Consumption>> getConsumptionData(
    String cups,
    String distributorCode,
    String startDate,
    String endDate,
    int pointType) async {
```

```
    var prefs = await SharedPreferences.getInstance();
    String? token = prefs.getString('datadisToken');

    final response = await http.get(
      Uri.parse(
          'https://datadis.es/api-private/api/get-consumption-
data?cups=$cups&distributorCode=$distributorCode&startDate=${startDate}&endDate=${endDate}&me
asurementType=0&pointType=$pointType'),
      headers: {
        'Content-Type': 'application/json',
        "Accept": "application/json",
        'Authorization': 'Bearer $token',
      },
    );

    if (response.statusCode == 200) {
      final jsonResponse = jsonDecode(response.body);
      final consumptions =
          List<Map<String, dynamic>>.from(jsonResponse);
          consumptions.map((e) => print(e));
      return consumptions.map((json) => Consumption.fromJson(json)).toList();
    } else {
      throw Exception('Failed to load consumptions (${response.body})');
    }
  }

  ///Obtener potencias de un suministro
  static Future<List<Power>> getMaxPower(String bearerToken, String cups,
      String distributorCode, String startDate, String endDate) async {
    final uri =
        Uri.parse('https://datadis.es/api-private/api/get-consumption-data');
    final headers = {'Authorization': 'Bearer $bearerToken'};
    final queryParams = {
      'cups': cups,
      'distributorCode': distributorCode,
      'startDate': startDate,
      'endDate': endDate
    };

    final response = await http.get(uri.replace(queryParameters: queryParams),
        headers: headers);
    if (response.statusCode == 200) {
      final jsonResponse = jsonDecode(response.body);
      final powers = List<Map<String, dynamic>>.from(jsonResponse['powers']);
      return powers.map((json) => Power.fromJson(json)).toList();
    } else {
      throw Exception('Failed to load consumptions');
    }
  }
}
```

Tabla 24 - Código definición métodos API

```
//Obtención de los datos a través de la API
ContractDetail contractDetail = await API.getContractDetail(cups, distributorCode);
//Obtención de los valores a insertar
    var value = {
     'cups': contractDetail.cups,
     'distributor': contractDetail.distributor,
     'marketer': contractDetail.marketer,
     'tension': contractDetail.tension,
     'accessFare': contractDetail.accessFare,
     'province': contractDetail.province,
     'municipality': contractDetail.municipality,
     'postalCode': contractDetail.postalCode,
     'contractedPowerkWMin': contractDetail.contractedPowerkWMin,
     'contractedPowerkWMax': contractDetail.contractedPowerkWMax,
     'timeDiscrimination': contractDetail.timeDiscrimination,
     'startDate': contractDetail.startDate,
    };
//Inserción de los datos en la tabla
    database.insert('contracts', value);
```

Tabla 25 - Código llamada a API

III. Código fuente construcción de clases

```
class Supply {
  String? address;
  String? cups;
  String? postalCode;
  String? province;
  String? municipality;
  String? distributor;
  String? validDateFrom;
  String? validDateTo;
  int? pointType;
  String? distributorCode;

  Supply(
      {this.address,
        this.cups,
        this.postalCode,
        this.province,
        this.municipality,
        this.distributor,
        this.validDateFrom,
        this.validDateTo,
        this.pointType,
        this.distributorCode});

  Supply.fromJson(Map<String, dynamic> json) {
    address = json['address'];
    cups = json['cups'];
    postalCode = json['postalCode'];
    province = json['province'];
```

```
    municipality = json['municipality'];
    distributor = json['distributor'];
    validDateFrom = json['validDateFrom'];
    validDateTo = json['validDateTo'];
    pointType = json['pointType'];
    distributorCode = json['distributorCode'];
  }

  Map<String, dynamic> toJson() {
    final Map<String, dynamic> data = new Map<String, dynamic>();
    data['address'] = this.address;
    data['cups'] = this.cups;
    data['postalCode'] = this.postalCode;
    data['province'] = this.province;
    data['municipality'] = this.municipality;
    data['distributor'] = this.distributor;
    data['validDateFrom'] = this.validDateFrom;
    data['validDateTo'] = this.validDateTo;
    data['pointType'] = this.pointType;
    data['distributorCode'] = this.distributorCode;
    return data;
  }
}
```

Tabla 26 - Código clase Supply

IV. Código fuente construcción de componentes

```
class AppBars extends StatefulWidget implements PreferredSizeWidget {
  final String? title;

  AppBars({this.title});

  @override
  _AppBarsState createState() => _AppBarsState();

  @override
  Size get preferredSize => Size.fromHeight(kToolbarHeight);
}

class _AppBarsState extends State<AppBars> {
  @override
  Widget build(BuildContext context) {
    return AppBar(
      title: Text(widget.title ?? ''),
      centerTitle: true,
      actions: <Widget>[
      PopupMenuButton(itemBuilder: (context) {
        return Language.languageList().map((e) => PopupMenuItem(value: e,child: Row(
            mainAxisAlignment: MainAxisAlignment.spaceAround,
            children: [
              Text(e.flag, style: TextStyle(fontSize: 30),),
              Text(e.name)
```

```
          ],
        ))).toList();
    },
    icon: const Icon(Icons.language, color: Colors.white,),
    onSelected: (lang) async {
      print((lang as Language).languageCode);
        final provider = Provider.of<LocaleProvider>(context, listen: false);
        provider.setLocale(Locale((lang as Language).languageCode));
        var preferences = await SharedPreferences.getInstance();
        preferences.setString("language", (lang as Language).languageCode);

      },
    ),
    IconButton(
      icon: const Icon(Icons.notifications),
      onPressed: () => print('notificaciones'),
    ),
    IconButton(
      icon: const Icon(Icons.logout),
      onPressed: () async {
        final prefs = await SharedPreferences.getInstance();
        prefs.clear();
        Navigator.of(context).pushReplacement(MaterialPageRoute(
          builder: (context) => LoginScreen(),
        ));
      },
    ),
  ],
  );
 }
}
```

Tabla 27 - Código creación de componente AppBar persnalizado

V. Código fuente proveedor para gestión de los idiomas

```
class LocaleProvider extends ChangeNotifier{
  Locale _locale = Locale('es');
  Locale get locale => _locale;

  void setLocale(Locale locale){
    LocalizationsDelegate<AppLocalizations> delegate = AppLocalizations.delegate;
    if(!delegate.isSupported(locale)) return;

    _locale = locale;
    notifyListeners();
  }

  void clearLocale(){
    _locale = const Locale('es');
    notifyListeners();
  }
}
```

VI. Código fuente del fichero pubspec.yaml

```yaml
name: tfgproyecto
description: A new Flutter project.
# The following line prevents the package from being accidentally published to
# pub.dev using `flutter pub publish`. This is preferred for private packages.
publish_to: 'none' # Remove this line if you wish to publish to pub.dev
version: 1.0.0+1

environment:
  sdk: ">=2.17.0 <3.0.0"
dependencies:
  cupertino_icons: ^1.0.2
  http: ^0.13.5
  flutter_login: ^4.1.1
  sqflite: ^2.2.5
  path: ^1.8.2
  shared_preferences: ^2.0.18
  charts_flutter: ^0.12.0
  table_calendar: ^3.0.3
  encrypt: ^5.0.1
  flutter_dotenv: ^5.0.0
  provider: ^6.0.5
  flutter_localizations:
    sdk: flutter
  intl: ^0.17.0-nullsafety.2

dev_dependencies:
  flutter_test:
    sdk: flutter
  flutter_lints: ^2.0.0

# The following section is specific to Flutter packages.
flutter:
  generate: true
  uses-material-design: true

  # To add assets to your application, add an assets section, like this:
  assets:
  - assets/logo.png
  - assets/logo_old.png
```

VII. Código fuente fichero traducción i10n

```
{
    "title": "Ten el control de tus consumos en tus manos",
    "language": "Español",
    "@language":{
        "description": "El idioma actual"
    },
```

```
        "home": "Inicio",
        "supplies": "Suministros",
        "profile": "Perfil",

        "minimum": "Mínimo",
        "maximum": "Máximo",
        "actual": "Actual",

        "price_title": "Precios del {price}",
        "price": "Precio {price} €/kWh",
        "@price":{
           "type": "text",
           "placeholders": {
              "price": {}
           }
    },
}
```

Bibliografía

[1] Energía y sociedad, «El proceso de liberalización y separación de actividades reguladas,» [En línea]. Available: https://www.energiaysociedad.es/manual-de-la-energia/4-1-el-proceso-de-liberalizacion-y-separacion-de-actividades-reguladas/. [Último acceso: Junio 2023].

[2] Endesa, «endesa.com,» [En línea]. Available: https://www.endesa.com/es/blog/blog-de-endesa/horarios-luz-valle-punta-llano. [Último acceso: 13 junio 2023].

[3] «AutoSolar,» [En línea]. Available: https://cdn.autosolar.es/images/blog/mapa-distribuidoras-electricas.png.

[4] «datadis,» [En línea]. Available: datadis.es.

[5] SCRUM, «scrum.org,» [En línea]. Available: https://scrumorg-website-prod.s3.amazonaws.com/drupal/inline-images/2023-02/screenshot_2023-02-14_at_8.36.08_am.png.

[6] V. Diví, «inLab,» [En línea]. Available: https://inlab.fib.upc.edu/es/blog/que-es-el-lenguaje-de-programacion-dart. [Último acceso: junio 2023].

[7] Canonical, «Canonical Snapcraft,» 4 Abril 2019. [En línea]. Available: https://snapcraft.io/blog/visual-studio-code-launches-as-a-snap. [Último acceso: 5 Junio 2023].

[8] Wikipedia, «Wikipedia.org,» [En línea]. Available: https://es.wikipedia.org/wiki/Archivo:Client-server-model.svg. [Último acceso: 5 Junio 2023].

[9] Flutter, «Flutter,» [En línea]. Available: https://docs.flutter.dev/ui/widgets-intro.

[10] BOE Legislación LOPD, «Boletín Oficial del Estado,» 5 Diciembre 2018. [En línea]. Available: boe.es.

[11] Universidad de Las Palmas de Gran Canaria, «Universidad de Las Palmas de Gran Canaria,» [En línea]. Available: https://www2.ulpgc.es/archivos/plan_estudios/4008_40/ObjetivosyCompetenciasdelGII.pdf.

[12] H. Hernandez, Artist, *Hexágono teoría del color.* [Art]. SlashMobility.

[13] Energigreen, «energigreen,» [En línea]. Available: https://www.energigreen.com/mercado-electrico-espanol/.

[14] A. E. Perejón, «Theconversation,» [En línea]. Available: https://theconversation.com/por-que-cambia-tanto-el-precio-de-la-electricidad-en-espana-154419.

[15] Guardia Civil & UC3M, «biblioteca.guardiacivil.es,» [En línea]. Available: https://biblioteca.guardiacivil.es/cgi-bin/koha/opac-retrieve-file.pl?id=ae6334d810a192ae4635dac65c4c3c6f.

[16] Endesa, «endesa,» [En línea]. Available: https://www.endesa.com/es/la-cara-e/red-electrica/contador-inteligente.

[17] Wikipedia, «Wikipedia: La enciclopedia libre,» [En línea]. Available: https://upload.wikimedia.org/wikipedia/commons/1/1c/Cliente-Servidor.png.

[18] RedHat, «redhat,» 8 Mayo 2020. [En línea]. Available: https://www.redhat.com/en/topics/api/what-is-a-rest-api#:~:text=A%20REST%20API%20(also%20known,interaction%20with%20RESTful%20web%20services..

[19] Flutter, «Flutter docs,» [En línea]. Available: https://docs.flutter.dev.

[20] F. Campus, «Flutter Campus,» [En línea]. Available: https://www.fluttercampus.com/guide/42/how-to-open-url-in-external-browser-in-flutter-app/.

[21] Stackoverflow, «stackoverflow,» [En línea]. Available: https://stackoverflow.com/questions/65764725/return-string-from-a-future-function.

[22] Flutter, «api.flutter.dev,» [En línea]. Available: https://api.flutter.dev/flutter/widgets/Navigator-class.html.

[23] pub.dev, «pub.dev,» [En línea]. Available: https://pub.dev/packages.

[24] REE, «REData API,» [En línea]. Available: https://www.ree.es/en/apidatos.

[25] M. Papageorgiou, «mariapapag,» 4 Octubre 2019. [En línea]. Available: https://mariapapag.medium.com/how-to-deploy-a-flutter-app-to-an-ios-device-48b286d921d3.

[26] DRIFT, «drift.simnbinder,» [En línea]. Available: https://drift.simonbinder.eu/docs/advanced-features/isolates/.

[27] C. Loguercio, «System Weakness,» 20 Marzo 2022. [En línea]. Available: https://systemweakness.com/how-to-store-login-credentials-the-right-way-in-flutter-857ba6e7e96d.

[28] P. Darji, «Log Rocket: Frontend Analytics,» 26 Mayo 2021. [En línea]. Available: https://blog.logrocket.com/how-to-build-a-bottom-navigation-bar-in-flutter/.

[29] R. Palankar, «Proto Coders Point,» 2 Agosto 2021. [En línea]. Available: https://protocoderspoint.com/flutter-pass-data-between-screens/.

[30] Wikipedia, «Wikipedia.org,» [En línea]. Available: https://es.wikipedia.org/wiki/JSON. [Último acceso: Mayo 2023].

[31] Wikipedia, «Wikipedia.org,» [En línea]. Available: https://es.wikipedia.org/wiki/Inteligencia_empresarial. [Último acceso: Mayo 2023].


ULPGC
Universidad de
Las Palmas de
Gran Canaria
Escuela de
Ingeniería
Informática