*Alejandro García Moreno 2ºDAM*

Memoria TRABAJO FIN DE GRADO



***Índice:***

1. Introducción
   1. Ideas planteadas
   2. Idea final
      1. Resumen del proyecto
      2. Motivación
      3. Objetivos propuestos
2. Tecnologías y herramientas utilizadas
   1. Android Studio
   2. Usar Git como repositorio de código fuente
   3. Sawagger
   4. SQL Server management studio
   5. SQLite
   6. Postman
   7. Paquete Office
3. Desarrollo
   1. Descripción funcional
      1. Introducción
      2. Control de usuario
      3. Sincronización de datos
      4. Introducción de fichas de clientes
   2. Diseño de pantallas y menús
   3. Diagramas y explicaciones
      1. Login
      2. Nuevas fichas
      3. Fichas Activas
      4. Sincronización
4. Pruebas
   1. Pruebas unitarias
   2. Pruebas de integración
5. Producto final
   1. Resultado final
   2. Conclusiones
   3. Vías futuras
   4. Bibliografía

***1. Introducción***

Al inicio de este Trabajo Final de Grado (TFG), era fundamental tener un proyecto claro antes de comenzar con las prácticas. A lo largo del curso, dediqué tiempo a considerar varias ideas sobre qué realizar.

Primero, reflexioné sobre la plataforma en la que llevaría a cabo mi trabajo, considerando opciones como una página web, una aplicación móvil o alguna otra idea diferente. Finalmente, me decidí por programar una aplicación móvil, ya que esta fue la opción que más disfruté aprender durante el año en la Cámara de Comercio, en el grado de Desarrollo de Aplicaciones Multiplataforma (DAM).

Además de parecerme lo más divertido y entretenido, consideré que una aplicación móvil tendría mucha más utilidad que otras opciones, debido a que todos los usuarios tienen un dispositivo móvil consigo. Las aplicaciones móviles son esenciales en la actualidad, ya que proporcionan acceso rápido y directo a los servicios y funcionalidades que ofrecen. Por otro lado, las páginas web suelen requerir pantallas más grandes, siendo más útiles en portátiles u ordenadores. Sin embargo, no todo el mundo tiene acceso constante a estos dispositivos, y buscar o abrir una página web en un celular puede resultar incómodo debido al tamaño reducido de las pantallas al mismo tiempo que se pierde tiempo en buscar la página web en el buscador de internet. En cambio, una aplicación móvil permite un acceso directo con un solo toque, ahorrando tiempo al usuario y satisfaciendo sus necesidades de manera más eficiente. Por todas estas razones, decidí crear una aplicación móvil para mi proyecto.

***1.1 Ideas planteadas***

Una vez decidido por qué plataforma decantarme, el siguiente paso era definir en qué iba a consistir el proyecto. Este proceso de conceptualización requirió una reflexión profunda y una lluvia de ideas para explorar todas las posibilidades. Quería asegurarme de elegir una idea que no solo fuera viable desde un punto de vista técnico, sino que también ofreciera un valor añadido a los usuarios potenciales. Con esto en mente, comencé a considerar diferentes tipos de aplicaciones, evaluando cada una en términos de originalidad, utilidad y atractivo para el mercado. Pasé tiempo investigando tendencias actuales, analizando necesidades no satisfechas y reflexionando sobre las áreas en las que podría innovar. De esta manera, me permití soñar con diversas posibilidades antes de centrarme en las opciones más prometedoras. Algunas de las ideas que surgieron durante este proceso fueron:

* App de un videojuego:

Primero pensé en realizar un videojuego nuevo, novedoso y divertido, dirigido principalmente a un público adolescente. Mi intención era crear un juego que fuera accesible y fácil de entender, de modo que cualquier persona, independientemente de su experiencia previa con los videojuegos, pudiera disfrutarlo. No llegué a definir una idea concreta del videojuego, pero sabía que quería que fuese simple y entretenido, siguiendo la línea de juegos populares como Geometry Dash. Este tipo de juegos se caracteriza por su jugabilidad adictiva y su capacidad para captar el interés de los jugadores rápidamente, gracias a sus mecánicas de juego sencillas pero desafiantes. Mi objetivo era desarrollar un juego que proporcionara una experiencia similar, donde los usuarios pudieran disfrutar de partidas rápidas y emocionantes, manteniéndose enganchados y deseando mejorar sus habilidades con cada intento.

* App de un calendario de empresa

Una de las ideas que consideré fue la de desarrollar una aplicación de calendario específicamente diseñada para empresas. La idea central era crear una herramienta que permitiera a las empresas controlar y gestionar el horario de todos sus trabajadores de manera eficiente. Aunque muchas empresas ya cuentan con aplicaciones propias para este propósito, mi objetivo era diseñar una aplicación que fuera universalmente funcional, es decir, adaptable a cualquier tipo de empresa sin importar su tamaño o sector.

La aplicación permitiría a los administradores editar y gestionar el calendario de los empleados, asignando turnos y horarios de manera sencilla. Además, cada usuario tendría la capacidad de solicitar días libres, como vacaciones o permisos especiales, directamente a través de la aplicación. Estas solicitudes aparecerían en el calendario y quedarían pendientes hasta ser confirmadas por el administrador o jefe correspondiente. Una vez aprobadas, los cambios se reflejarían automáticamente en el calendario de todos los usuarios, garantizando una comunicación clara y eficiente dentro de la empresa. Esta funcionalidad no solo facilitaría la organización del personal, sino que también mejoraría la transparencia y la satisfacción de los empleados al tener una herramienta confiable para gestionar sus horarios y solicitudes.

* App de gestión de tareas personales:

Otra idea fue desarrollar una aplicación para la gestión de tareas personales. Esta app permitiría a los usuarios organizar sus actividades diarias, semanales o mensuales de manera eficiente. Podrían crear listas de tareas, establecer recordatorios, y asignar prioridades a cada actividad. Además, la aplicación podría incluir funcionalidades de seguimiento del progreso y análisis de la productividad, ofreciendo estadísticas y sugerencias para mejorar la gestión del tiempo. Esta herramienta sería ideal para estudiantes, profesionales y cualquier persona que desee optimizar su tiempo y aumentar su productividad.

* App de recetas rápidas y sencillas:

Esta aplicación estaría enfocada en ofrecer recetas fáciles y rápidas de preparar, ideal para personas con poco tiempo o poca experiencia en la cocina. Los usuarios podrían buscar recetas según diferentes criterios como ingredientes disponibles, tiempo de preparación, tipo de comida (desayuno, almuerzo, cena, postre), o preferencias dietéticas (vegetariano, sin gluten, etc.). Cada receta incluiría una lista de ingredientes, instrucciones paso a paso, y fotos o videos demostrativos. Además, la app podría tener una función para generar una lista de compras basada en las recetas seleccionadas. Esta aplicación sería especialmente útil para estudiantes, profesionales ocupados, o cualquier persona que busque simplificar su rutina de cocina diaria.

***1.2 Idea final***

Finalmente, tomé la decisión de optar por una aplicación para Tablet que me recomendó mi tía con el propósito de poder asistirla en su trabajo. Me pareció una herramienta muy práctica y, al mismo tiempo, un desafío que estaba dispuesto a afrontar, considerándola, así como mi primer cliente.

**1.2.1 Resumen del proyecto**

El proyecto consiste en desarrollar una aplicación de gestión de relaciones con clientes (CRM) específicamente diseñada para la peluquería de mi tía. La aplicación permitirá gestionar de manera eficiente a los clientes que pasan por la peluquería, registrando los cortes y servicios que se realizan y calculando automáticamente el precio total, incluyendo el IVA y posibles descuentos. Las principales funciones de la app incluyen:

* **Inicio de sesión:** La aplicación contará con una pantalla de inicio de sesión. El usuario predeterminado estará creado en la base de datos, y para acceder a la app deberá ingresar su correo electrónico y contraseña. Una vez iniciado sesión, la aplicación no volverá a solicitar estas credenciales a menos que el usuario elija cerrar sesión.
* **Gestión de fichas de clientes**: La app permite crear y actualizar fichas de clientes, tanto para clientes existentes como para nuevos clientes. En estas fichas se registran los detalles de los servicios prestados y los cortes de pelo realizados.
* **Cálculo automático de precios**: La aplicación calcula automáticamente el precio total de los servicios, añadiendo el IVA y aplicando cualquier descuento correspondiente, facilitando así la gestión de cobros y facturación.
* **Visualización de fichas por fecha**: Los usuarios pueden ver todas las fichas de clientes creadas en una fecha específica, simplemente seleccionando la fecha en un calendario integrado en la aplicación.
* **Sincronización de datos**: La aplicación incluye una función de sincronización que permite conectar y sincronizar mis datos con los de la base de datos principal de la página web de la peluquería. Esta página web ofrece opciones adicionales, pero la aplicación para Tablet se centra exclusivamente en la gestión de las fichas de clientes, por lo que, deberé crear una base de datos adicional para la app centrada solamente en los datos necesarios para dichas gestiones. La sincronización se realiza mediante una API que permite acceder de manera segura a la base de datos oficial.

En resumen, esta aplicación no solo facilita la gestión diaria de los clientes en la peluquería, sino que también garantiza la integración y sincronización con la infraestructura digital existente, mejorando la eficiencia y la organización del negocio.

**1.2.2 Motivación**

La motivación detrás de este proyecto se basa en que se trata de un caso real con un propósito tangible y beneficioso. Este TFG no es solo un ejercicio académico, sino una solución práctica que ayudará directamente a un familiar. Mi tía, que posee una peluquería, enfrenta dificultades debido a la falta de Wifi en su establecimiento. Actualmente, no puede llevar su portátil para realizar las fichas de los clientes, por lo que se ve obligada a escribir todo a mano durante su jornada laboral y luego transcribir la información al ordenador en su casa. Este proceso duplica su trabajo y le resta tiempo valioso que podría dedicar a otras actividades.

En cambio, ella posee una Tablet con datos móviles, la cual podría ser una herramienta ideal para mejorar su flujo de trabajo. La facilidad de uso y la accesibilidad de la Tablet permitirán a mi tía optimizar su tiempo y recursos. Podrá registrar los cortes y servicios realizados, calcular automáticamente los precios con IVA y descuentos, y mantener todas las fichas de las clientes actualizadas en tiempo real. Además, cuando esté en casa o en cualquier lugar con acceso a internet, solo necesitará pulsar el botón de sincronización para actualizar la base de datos principal. De esta manera, los datos se integrarán automáticamente en su página web, donde podrá realizar funciones adicionales según sea necesario.

Este proyecto no solo tiene la motivación de facilitar el trabajo diario de mi tía, sino también de demostrar cómo una solución tecnológica bien diseñada puede mejorar significativamente la eficiencia operativa de un negocio. La motivación personal de ayudar a un familiar y mejorar su calidad de vida, combinada con el desafío técnico y la oportunidad de aplicar los conocimientos adquiridos durante el grado, hace que este TFG sea especialmente significativo y gratificante.

**1.2.3 Objetivos propuestos**

El desarrollo de esta aplicación tiene varios objetivos clave que buscan asegurar su funcionalidad, fiabilidad y estética, de manera que satisfaga las necesidades específicas de mi tía y cumpla con los estándares de calidad esperados. A continuación, se detallan los principales objetivos propuestos:

* Funcionalidad completa y precisa:
  + Cálculo automático del total con IVA y descuentos: La aplicación debe ser capaz de calcular automáticamente el precio total de los servicios ofrecidos, incluyendo el IVA y aplicando los descuentos pertinentes. Este cálculo debe ser preciso y reflejarse de manera clara en la interfaz de usuario.
  + Gestión eficiente de fichas de clientes: La app permitirá la creación y actualización de fichas de clientes, tanto para nuevos como para los ya existentes. Esta funcionalidad debe ser intuitiva y rápida, facilitando la gestión diaria de la información de los clientes.
  + Sincronización de datos: Es esencial que la función de sincronización funcione sin errores, permitiendo que los datos ingresados en la aplicación se sincronicen correctamente con la base de datos de la página web. Esto garantizará que toda la información esté actualizada y accesible tanto en la app como en la web.
* Fiabilidad y ausencia de errores:
  + La aplicación debe ser robusta y funcionar sin errores. Durante el desarrollo y las pruebas, se debe poner un énfasis especial en la detección y corrección de cualquier fallo que pueda surgir, asegurando una experiencia de usuario fluida y sin interrupciones.
* Cumplimiento con los requisitos estéticos:
  + Diseño visual: La estética de la aplicación debe alinearse con los requisitos especificados por mi tía, manteniendo la coherencia con la página web existente. Esto incluye un esquema de colores predominantemente oscuros, con el negro y los grises como colores principales, y el blanco utilizado para la cabecera y algunas otras opciones secundarias.
  + Interfaz de usuario intuitiva: Además de cumplir con los requisitos de color, la interfaz debe ser intuitiva y fácil de navegar, proporcionando una experiencia de usuario agradable y eficiente.
* Satisfacción del cliente:
  + El objetivo principal es que mi tía se sienta orgullosa y contenta con el resultado final. La aplicación debe no solo cumplir con sus necesidades prácticas, sino también ser algo que le guste usar. Su satisfacción es una medida crucial del éxito de este proyecto.

En resumen, los objetivos propuestos para este proyecto son garantizar que la aplicación sea completamente funcional, libre de errores y estéticamente agradable, cumpliendo con las especificaciones proporcionadas por mi tía. La finalidad última es crear una herramienta que facilite su trabajo diario y que le brinde una experiencia satisfactoria y eficiente.

***2. Tecnologías y herramientas utilizadas***

En el desarrollo de este proyecto se utilizaron diversas tecnologías y herramientas que resultaron esenciales para alcanzar los objetivos propuestos. A continuación, se detallan las principales herramientas empleadas, comenzando por la más importante:

**2.1. Android Studio**

Android Studio es la herramienta principal y más crucial utilizada en este proyecto. Este entorno de desarrollo integrado (IDE) es específicamente diseñado por Google para el desarrollo de aplicaciones Android. Su importancia radica en que proporciona todas las funcionalidades necesarias para crear, probar y depurar aplicaciones para dispositivos con sistema operativo Android.

* Características principales:
  + Editor de código avanzado: Ofrece un editor de código potente con resaltado de sintaxis, autocompletado y refactorización, lo cual facilita la escritura y el mantenimiento del código.
  + Diseño de interfaz gráfica: Incluye un diseñador visual que permite crear y ajustar interfaces de usuario mediante un sistema de arrastrar y soltar componentes, simplificando el diseño de la interfaz.
  + Emulador de Android: Android Studio cuenta con un emulador integrado que permite probar la aplicación en diferentes dispositivos virtuales con varias versiones de Android, asegurando su correcto funcionamiento en múltiples configuraciones.
  + Depuración y pruebas: Herramientas avanzadas de depuración y pruebas que ayudan a identificar y corregir errores, garantizando que la aplicación funcione de manera óptima.
  + Gestión de dependencias: Utiliza Gradle como sistema de construcción, lo que facilita la gestión de dependencias y la automatización de tareas de construcción.

Android Studio no solo proporciona el entorno adecuado para el desarrollo de la aplicación, sino que también integra todas las herramientas necesarias para gestionar cada etapa del ciclo de vida del desarrollo de software, desde la concepción inicial hasta la implementación final. Esto hace que sea la herramienta fundamental en este proyecto, permitiendo la creación de una aplicación eficiente, funcional y estéticamente agradable.

En los siguientes apartados, se describirán otras herramientas y tecnologías complementarias que fueron utilizadas junto con Android Studio para completar el desarrollo de esta aplicación.

**2.2 Git**

Git se utilizó como el sistema de control de versiones y repositorio de código fuente para este proyecto. Git es una herramienta esencial en el desarrollo de software, ya que permite gestionar y controlar los cambios en el código fuente de manera eficiente, facilitando la colaboración y asegurando la integridad del proyecto a lo largo del tiempo.

En este proyecto, me uní al repositorio de mi tío, quien creó la página web, como contribuidor, el cual está dedicado a "Mika", el nombre de la peluquería. Este repositorio contiene todos los archivos necesarios para la creación y mantenimiento de la página web de la peluquería. Al unirme como colaborador, tuve acceso a todo el historial de versiones y cambios realizados en el proyecto, lo que me permitió trabajar de manera integrada y coordinada con los archivos existentes.

Mi trabajo se centró en desarrollar específicamente el apartado de "MikaAndroid" desde cero, asegurando que la aplicación móvil estuviera completamente integrada con el ecosistema digital de la peluquería. De esta manera, todo el código fuente, tanto de la página web como de la aplicación móvil, se mantiene en un único repositorio, facilitando la gestión y el mantenimiento del proyecto en el futuro.

* **Ventajas de usar Git en este proyecto**:
  + **Control de versiones**: Permite llevar un registro detallado de todos los cambios realizados en el código, facilitando la identificación y corrección de errores.
  + **Colaboración**: Facilita la colaboración con otros desarrolladores, permitiendo que múltiples contribuidores trabajen en diferentes partes del proyecto simultáneamente.
  + **Integración continua**: Al mantener todo el código en un solo repositorio, se facilita la integración continua y el despliegue, asegurando que tanto la página web como la aplicación móvil funcionen de manera coherente y sin conflictos.
  + **Seguridad y respaldo**: Git proporciona un sistema de respaldo robusto, asegurando que el código fuente esté protegido y sea recuperable en caso de cualquier eventualidad.

Para más detalles sobre el repositorio y el proyecto, se puede visitar el siguiente enlace: [GitHub - Mika Repository](https://github.com/jma-ludiverso/Mika).

En resumen, Git ha sido una herramienta fundamental para la organización y el desarrollo de este proyecto, permitiendo una gestión eficiente del código y facilitando la colaboración. La integración del apartado "Mika Android" en el repositorio principal asegura que toda la infraestructura digital de la peluquería esté unificada y bien organizada.

**2.3 Swagger**

Es una herramienta esencial utilizada en este proyecto para la definición y visualización de APIs. Específicamente, ha sido fundamental para entender la API que permite la sincronización de datos entre la aplicación móvil y la base de datos de la página web de la peluquería. La documentación detallada proporcionada por Swagger incluye todas las tablas de datos necesarias y los endpoints que se deben utilizar, facilitando así el desarrollo y la integración de la aplicación.

* **Acceso a la documentación de la API:** En ella se proporciona una interfaz visual intuitiva que permite explorar y probar los distintos endpoints de la API. A través de esta interfaz, pude ver claramente los métodos disponibles (GET, POST, PUT, DELETE), los parámetros requeridos y los posibles resultados de cada llamada.
* **Definición clara de los datos**: La herramienta permite visualizar las estructuras de las tablas de datos necesarias para el proyecto, asegurando que todos los campos y tipos de datos están correctamente definidos y alineados con los requisitos de la aplicación. Esto incluye información detallada sobre los clientes, los servicios prestados, los precios, el IVA y los descuentos aplicables.
* **Pruebas y validación:** Una de las grandes ventajas que tiene es la posibilidad de realizar pruebas directamente desde la interfaz. Esto me permitió validar las llamadas a la API, asegurando que los datos se sincronizan correctamente y que la aplicación interactúa de manera efectiva con la base de datos. Pude verificar que la creación de nuevas fichas de clientes, la actualización de información existente y la sincronización de datos funcionan sin problemas.
* **Facilitación de la integración:** Esta simplifica enormemente la integración de la API con la aplicación móvil. Al disponer de una documentación exhaustiva y de una interfaz de prueba, pude desarrollar y ajustar el código de la aplicación con mayor precisión y confianza. Esto asegura que la funcionalidad de sincronización sea robusta y fiable, permitiendo que los datos ingresados en la aplicación se reflejen correctamente en la base de datos principal.

Para más detalles y acceso a la documentación completa de la API utilizada, se puede visitar el siguiente enlace: [Swagger API Documentation](https://test.ludiverso.com/swagger/index.html).

En resumen, Swagger ha sido una herramienta crucial en este proyecto, proporcionando la documentación necesaria y la capacidad de prueba para integrar eficazmente la API. Esto ha asegurado que la aplicación móvil y la base de datos web estén perfectamente sincronizadas, cumpliendo con los requisitos funcionales y mejorando la eficiencia operativa de la peluquería.

**2.4 SQL Server management studio**

SQL Server Management Studio (SSMS) es una herramienta crítica utilizada en este proyecto para gestionar la base de datos principal de la peluquería. Esta base de datos, creada previamente para la aplicación web, recopila y almacena todos los datos necesarios para el funcionamiento de la página web de la peluquería, incluyendo información sobre clientes, servicios ofrecidos, precios y otros elementos esenciales.

Se ha utilizado durante el desarrollo de este proyecto como herramienta de visualización de la estructura y definición de las tablas y campos que era necesario replicar en SQLite.

* **Gestión de datos**: SSMS permite la creación, modificación y eliminación de tablas y registros, garantizando que la información de la página web esté siempre actualizada y sea precisa.
* **Consultas y análisis**: Facilita la ejecución de consultas SQL para obtener y manipular datos, permitiendo un análisis profundo y la generación de reportes necesarios.

Es importante señalar que mi contribución se centrará en recoger los datos necesarios de esta base de datos principal a través del Api para almacenarlos en otra base de datos específica para la aplicación Android. De esta manera, se asegura que la información esté correctamente sincronizada y accesible desde la aplicación móvil sin interferir con la base de datos original de la página web.

**2.5 SQLite**

SQLite es la base de datos utilizada para desarrollar la aplicación en la Tablet. Es una base de datos ligera y eficiente, ideal para aplicaciones móviles debido a su fácil integración y bajo consumo de recursos. En este proyecto, las tablas necesarias de la base de datos principal en SQL Server Management Studio fueron convertidas al formato SQLite. Estas tablas fueron seleccionadas específicamente para cubrir las funciones requeridas por la aplicación, garantizando que solo se incluyan los datos necesarios.

Las tablas utilizadas en la base de datos SQLite son:

* Empresas
* Salones
* Empleados
* Empleados comisiones
* Empleados servicios
* Fichas
* Fichas Lineas
* Servicios
* Clientes
* Clientes Historial

Estas tablas permiten gestionar toda la información relevante para las operaciones diarias de la peluquería, como la administración de empleados, servicios ofrecidos, y el historial de clientes, asegurando una integración completa y eficiente con la aplicación móvil.

**2.6 Postman**

Es una herramienta muy útil que se utilizó en este proyecto para solucionar dudas relacionadas con la sincronización del API. En el proceso de desarrollo de la aplicación móvil, surgieron preguntas y problemas específicos sobre cómo interactuar con la API de la página web de la peluquería. Postman permitió abordar estas cuestiones de manera eficiente y efectiva.

Una de las ventajas clave de Postman es su capacidad para realizar solicitudes HTTP a APIs y ver las respuestas en tiempo real, simplificando el proceso de identificación y resolución de errores.

**2.7 Paquete Office**

El paquete Office, en particular Microsoft Word y PowerPoint, desempeñó un papel crucial en varios aspectos del proyecto.

Primero, Microsoft Word se utilizó al principio del proyecto para anotar ideas, hacer una descripción funcional detallada y tener una guía clara de los objetivos y requisitos del proyecto. Esta, está descrita posteriormente en el punto **3.1 Descripción funcional**. Esto fue fundamental para establecer una dirección clara y asegurarse de que todas las partes interesadas estuvieran en la misma página desde el principio. Utilizar Word para documentar todas estas ideas y descripciones funcionales permitió mantener un registro organizado y fácilmente accesible de la visión general del proyecto, así como de los detalles específicos de cada función y requisito.

Además, Word también fue útil para redactar esta memoria que estoy explicando ahora. La capacidad de formato de texto, la inserción de imágenes y la organización de secciones facilitaron la creación de un documento claro y profesional que comunica eficazmente los detalles del proyecto.

Por otro lado, PowerPoint se utilizó para crear una vista del diseño solicitada por mi tía. Esta herramienta fue valiosa para visualizar y comunicar ideas de diseño de una manera clara y concisa. Al utilizar PowerPoint, se pudo crear una representación gráfica del diseño de la aplicación móvil, lo que permitió a tu tía comprender mejor cómo se vería y funcionaría la aplicación antes de comenzar el desarrollo real.

En resumen, el paquete Office, especialmente Microsoft Word y PowerPoint, desempeñó un papel esencial en el proceso de desarrollo de la aplicación móvil de la peluquería. Desde la documentación inicial de ideas y descripciones funcionales hasta la creación de vistas de diseño, estas herramientas proporcionaron las capacidades necesarias para planificar, comunicar y ejecutar el proyecto de manera efectiva.

***3. Desarrollo***

**3.1 Descripción funcional**

**3.1.1 Introducción**

Este punto describe las funcionalidades a cubrir para el desarrollo de la aplicación móvil para la gestión de una parte del día a día en Mika peluqueros.

La aplicación móvil, como ya hemos comentado anteriormente, debe estar optimizada para Tablet con sistema operativo Android 13 (“Tiramisú”), con un tamaño de pantalla aproximado de 1920 x 1200. La aplicación móvil completará el desarrollo actual de la aplicación web existente, permitiendo el trabajo en modo desconectado para aquellos momentos en los que la Tablet no tenga una conexión a internet disponible.

De todas las funcionalidades existentes en la aplicación web, la aplicación móvil permitirá el trabajo en modo desconectado para la introducción de fichas de clientes en el día a día de trabajo normal. Posteriormente, la aplicación deberá poder volcar toda esa información al servidor con un toque utilizando una conexión a internet disponible.

Inicialmente, el archivo .apk de la aplicación móvil estará disponible para su descarga en la propia web de la peluquería.

**3.1.2 Control de usuario**

Al abrir la aplicación móvil, ésta deberá detectar si el usuario está logado en la APP. En el caso en el que el usuario no se haya logado aún, o haya decidido cerrar su sesión, la APP mostrará una pantalla para la introducción del nombre de usuario y contraseña. El login de usuario requerirá que la aplicación disponga de conexión a internet.

El usuario y la contraseña se validarán contra el servidor y deberán quedarse almacenados en la APP para futuros usos, hasta que el usuario decida cerrar sesión. Una vez que el usuario se haya logado con éxito, la aplicación obtendrá los datos de configuración necesarios para poder realizar la introducción de fichas (ver punto de sincronización de datos).

**3.1.3 Sincronización de datos**

La sincronización de datos se realizará de manera automática tras el login de usuario, o desde una opción de menú disponible para ello. Esta sincronización supondrá la recepción y el envío de información desde y hasta el servidor web.

La recepción de datos supondrá la actualización de la información existente en la base de datos local de la APP de las tablas comentadas anteriormente (Empresas, Salones, Empleados, Empleados comisiones, Empleados servicios, Fichas, Fichas líneas. Servicios, Clientes, Clientes Historial y Android Metadata).

El envío de datos desde la APP al servidor, supondrá el envío de los datos de cabecera de las fichas y de cada una de sus líneas, así como de los clientes que se hayan dado de alta nuevos.

El proceso de sincronización supondrá la actualización de los datos anteriores y el volcado de fichas y clientes al servidor, vaciando este contenido en la base de datos temporal de la aplicación móvil.

**3.1.4 Introducción de fichas de clientes**

La pantalla para almacenar las distintas fichas de clientes deberá permitir buscar en el listado de clientes de la aplicación un cliente por nombre. La pantalla de búsqueda de cliente tendrá una opción para introducir los datos de un nuevo cliente.

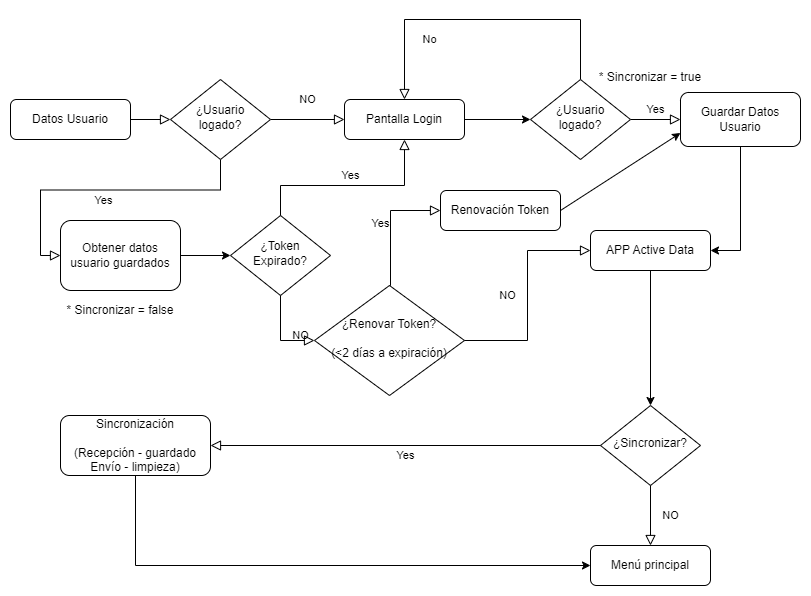
Una vez seleccionado o con el cliente genérico por defecto, se pasará a la pantalla de introducción de los servicios y productos de la ficha. La pantalla principal de la ficha de cliente, mostrará en la parte superior los datos de resumen de la ficha con sus importes, y en la parte inferior en listado de servicios y productos incluidos en la ficha. Esta pantalla tiene que permitir la inserción de un nuevo producto o servicio.

La pantalla de introducción de una nueva línea requerirá la indicación del código del empleado y del código del servicio o producto a insertar. La aplicación mostrará el precio configurado para el producto o servicio, pero éste podrá ser modificado manualmente por el usuario, con la posibilidad de indicar un porcentaje de descuento.

Una vez introducidas todas las líneas de la ficha, la cabecera de la ficha permitirá realizar el cobro, pudiendo ser éste en tarjeta o efectivo. En el caso de pago en efectivo la aplicación facilitará el cálculo del cambio a dar al cliente.

Desde el menú principal de la aplicación se podrá acceder al listado de fichas almacenadas en la Tablet, filtrando por fecha. Desde este listado podrá accederse a una de las fichas individuales para realizar cambios en ella.

**3.2 Diseño de pantallas y menús**

* 1. **Diagramas y explicaciones**
     1. **Login**

En el proceso de Login la aplicación comprueba si el usuario ha sido logado con anterioridad. Si el usuario no ha sido logado muestra la pantalla de Login para la introducción de los datos de acceso. Si los datos introducidos son correctos, la aplicación guarda los datos del usuario junto con el token de acceso al Api Web de donde se obtendrán y enviarán datos.



*Nota aclaratoria al código:*

*Para la comunicación con el Api se han implementado las dependencias: com.android.volley:volley:1.2.0 y com.google.code.gson:gson:2.10.1 ; para la realización de las llamadas al Api y para la serialización y deserialización de datos en formato json. La url base del api se encuentra configurada en el archivo Strings.*

*Para el manejo de datos con el Api se han creado varias clases en línea con la definición de datos proporcionados por el Api. Estas clases son: AuthenticateRequest, AuthenticateResponse y MikaWebUser.*

Si el usuario se ha logado con anterioridad, la aplicación obtiene los datos guardados y el token que fue asignado, comprobando si el token ha expirado. Si el token se encuentra expirado, la aplicación muestra la pantalla de login y sigue el proceso explicado anteriormente. Si el token es válido, se comprueba si está cercano a expirar. En caso afirmativo, se hace una llamada al Api para renovar el token y se actualizan los datos guardados del usuario.



*Nota aclaratoria al código:*

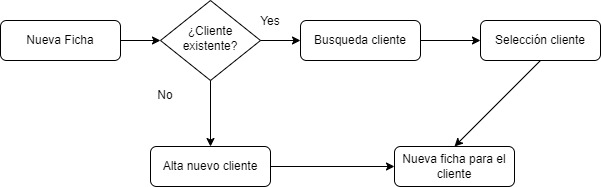
*En todas las llamadas posteriores al Api, la aplicación usa el token de acceso como un encabezado de la petición. Los datos del encabezado se completan en la función getHeaders().*

Una vez completado el proceso de login, la aplicación guarda en una clase estática los datos del usuario junto con su token para que estén disponibles en todo momento en la aplicación.



Como último paso, si es necesario sincronizar los datos con el servidor, se procede a llamar al api para obtener todos los datos necesarios para trabajar con la aplicación. Finalmente se muestra el menú principal.

* + 1. **Nuevas fichas**



El primer paso antes de crear una nueva ficha, consiste en la selección del cliente al que se le agregarán los servicios. La pantalla de selección de cliente permite buscar a un cliente existente de entre los clientes ya registrados obtenidos a través del Api. Si el cliente es nuevo, la pantalla permite darlo de alta en la App seleccionándolo automáticamente como cliente de la nueva ficha.





*Nota aclaratoria al código:*

*Para las operaciones de acceso a datos se ha creado una clase denominada DBManager en la que se realizan las operaciones básicas de creación de la base de datos (bbdd) a partir del archivo Mika.db ubicado en los assets del proyecto. También se realizan operaciones de apertura y cierre de la conexión, obtención de cursores de selección de datos y operaciones de inserción, actualización y borrado de datos. Junto con la clase DBManager se ha creado una clase DBStructure con las definiciones de los nombres de tablas y campos de la bbdd.*

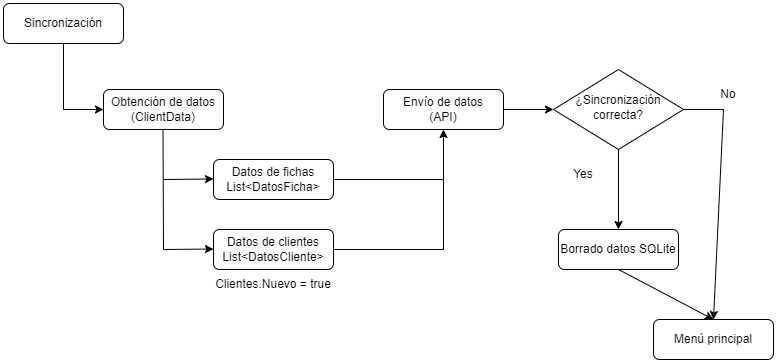
* + 1. **Fichas Activas**

La pantalla fichas activas permite consultar por fecha todas las fichas de cliente guardadas en la app móvil y que por tanto se encuentran pendientes de sincronizar con la web.

Las fichas existentes para una fecha se muestran en una tabla con el número de ficha, cliente, fecha e importe total. El numero de la ficha funciona a modo de enlace permitiendo al usuario entrar en la ficha seleccionada para realizar las modificaciones que sean necesarias.

La consulta de carga de datos se hace en la función cargaDatos(), donde se recorre el cursor de resultados obtenidos creando una nueva fila en la tabla existente en el diseño de la pantalla. Al pulsar en el botón de número de ficha se llama a la función cargaDatosNumeroFicha() en donde se obtienen los datos de la ficha con todos sus servicios y se realiza la llamada a la pantalla de modificación de ficha.

* + 1. **Sincronización**



En la pantalla de sincronización, previo al envío de datos del Api web, se obtienen los datos pendientes de envio de la bbdd SQLite. Por un lado, se obtienen los clientes dado de alta en la app que han sido marcados con un atributo nuevo en la bbdd, así como el listado de todas las fichas de cliente almacenadas con la app.

Si el proceso de envío de datos al Api finaliza correctamente, la aplicación procede a borrar los datos de los clientes y fichas que han sido enviados, retornando al menú principal de la aplicación.

Para el envío de datos al Api se han creado varias clases en línea con la definición de datos proporcionados por el Api. Estas clases son: ClientData, DatosFicha, DatosFichaLinea, DatosCliente, DatosClienteHistorial y DataResponse.



*Nota aclaratoria al código:*

*Función de captura de la respuesta de llamada al recurso setdata del Api. Si la propiedad success viene a true, la sincronización de datos se ha realizado correctamente y se realiza la llamada a la función DeleteData() de la clase DBManager.*

***4. Pruebas***

**4.1 Pruebas unitarias**

* Búsqueda de clientes

Se verifican los resultados obtenidos usando varias entradas de búsqueda diferentes. Se valida igualmente el funcionamiento y visualización correcta de los datos en la lista utilizando una clase adaptadora heredada de BaseAdapter.

* Inserción de clientes

Se comprueba que los datos de un nuevo cliente guardados a través de la aplicación se muestran después correctamente en el buscador. No se encuentran problemas en el proceso.

* Fichas – Nueva línea / modificación de línea / borrado de línea

Se prueba el correcto funcionamiento del dialog para introducir o modificar una línea en la ficha, probando la relación correcta con la clase de origen NuevaFicha.

Se prueba el correcto funcionamiento de los adaptadores creados para mostrar desplegables para los empleados y para los servicios. Se hacen varios ajustes en su visualización para que queden correctamente cuadrados. Se hacen varias correcciones en el código de los eventos asociados a la selección de un elemento de los desplegables anteriores.

Se comprueba la correcta creación en la pantalla de nueva ficha de una nueva línea y se comprueba la modificación de los datos en caso de editar una fila existente.

* Fichas – Líneas – Cálculo importes línea

Se comprueba que se realizan correctamente los calculas al modificar el porcentaje de descuento de una línea y/o su porcentaje de IVA. También se comprueba que los importes de la línea quedan correctos si se modifica directamente el importe final (total).

Se hacen varios ajustes en el formato de los números decimales, números con porcentajes, e importes con moneda €.

* Fichas – Cálculo importes resumen y grabación

Se hacen varias correcciones en el calculo resumido de los importes totales de la ficha según se van introduciendo, modificando, o eliminando líneas de la ficha. Se comprueba también el recalculo de los importes totales de la ficha y sus líneas en los casos en los que se indica un porcentaje de descuento general para toda la ficha. Se comprueba también no aplica descuentos a productos (requisito funcional del cliente).

Se comprueba la correcta grabación de todos los datos de la ficha y sus líneas, así como de todas las cantidades en € asociadas a porcentajes de descuento e IVA. Se corrige el formato de la fecha en la grabación de la ficha.

* Fichas activas – búsqueda por fecha

Se vuelve a encontrar un problema similar del formato de la fecha al realizar la consulta, aplicándose la misma solución para resolver el problema.

* Fichas activas – carga datos ficha existente

Se encuentran varios problemas al recuperar los datos de la bbdd local y al mostrar estos datos en la pantalla de nueva ficha. Requiere varias pruebas y ajustes tanto en la consulta a la bbdd como en el manejo de los datos en memoria de la ficha.

**4.2 Pruebas de integración**

* Proceso de login

Se prueba por primera vez el acceso al Api web, la serialización de datos en formato json y la deserialización de la respuesta obtenida. También se comprueba por primera vez la grabación de los datos obtenidos desde el Web Service en la bbdd local de la App.

Tras las primeras pruebas se detecta que es necesario incluir una fecha de expiración del token para que la App esté alineada con la seguridad de acceso de la Api web.

* Renovación de token

Se prueba por primera vez el uso del token de acceso a recursos protegidos en el Api. Se tienen que realizar varios ajustes en el manejo de la fecha de caducidad del token para comprobar si este se encuentra expirado o próximo a expirar.

* Sincronización datos – Recepción de datos

Se prueba la inserción y actualización masiva de datos en SQLite para la grabación de clientes, empleados, configuración de servicios y productos.

Se encuentran fallos en el código al realizar inserts de registros duplicados en la bbdd y se corrige para la realización de un update en los casos necesarios.

* Sincronización datos – Envío de datos

Se encuentran varios problemas en el formato del archivo json enviado al Api web, principalmente por el formato de fechas utilizado y por el nombrado incorrecto de alguna propiedad en la clase Clientdata. Se realizan varias pruebas obteniendo el contenido del json enviado a través del debug de Android Studio, y copiando dicho json en Postman para una más fácil compresión de los errores.