TFC - MEMORIA

Aplicación de reservas Booknow.

TFC- Desarrollo de aplicaciones para dispositivos móviles – Memoria, aplicación de reservas Booknow.

Javier Alonso Mayor
Ingeniería Técnica en Informática de Gestión

Consultores: Jordi Almirall Almirall e Ignasi Llorente Puchades

7 de Enero de 2013

"Le dedico este trabajo a mi padre Jesús Alonso Rilova que falleció víctima de un cáncer el 16-06-2006 y a toda mi familia"

Contenido

Ilustraciones

Introducción

En este documento se va hacer un repaso sobre todo el trabajo realizado para el proyecto de final de carrera dando una explicación de los diferentes puntos que lo forman, justificando la realización del mismo, realizando una crítica con propuestas de mejoras y explicando experiencias y aptitudes conseguidas.

Resumen

El desarrollo de aplicaciones móviles es una de las salidas profesionales con más auge en la actualidad y tal y como se mueve el mercado y la venta de dispositivos lo será también en el futuro.

En este proyecto realizamos una aplicación para el sistema operativo Android desarrollado por Google y utilizado por la mayoría de los fabricantes para sus dispositivos.

El objetivo de este proyecto ha sido realizar una aplicación que aprovecha muchas de las ventajas que nos ofrecen las tecnologías actuales y que nos permiten desarrollar aplicaciones que faciliten la vida de los usuarios y le den un valor añadido a los dispositivos que las utilizan, así como tener un primer contacto con el desarrollo de aplicaciones para dispositivos móviles y conseguir experiencias y aptitudes.

En el proyecto valoramos el tiempo y los recursos necesarios para la realización del proyecto, realizamos un estudio de diseño muy diferente al que utilizaríamos para otro tipo de aplicación basándonos en el usuario con DCU(diseño centrado en el usuario), y finalmente realizamos la implementación del proyecto utilizando en la tecnología Android los nuevos sistemas de persistencia en la nube y los dispositivos móviles con la intención de aprender y aprovechar todas sus ventajas teniendo siempre en mente al usuario final.

1. Guía de trabajo

1.1 Introducción

Este apartado nos ha servido de guía para el desarrollo del trabajo de final de carrera.

La idea de este apartado es que durante el desarrollo nos sirva como el guión del trabajo que tenemos que realizar.

En este apartado vamos a ver una pequeña especificación del proyecto y las diferentes fases que llevaremos a cabo durante el desarrollo, indicaremos que documentos entregaremos en cada fase y los objetivos de las mismas.

Para el desarrollo de las fases de este proyecto nos basaremos en un modelo en cascada combinado con un diseño centrado en el usuario.

1.2 Objetivo y alcance del proyecto:

El objeto de este proyecto es realizar una aplicación para dispositivos Android que nos permitirá realizar reservas para eventos de nuestra vida cotidiana como acudir a la peluquería, o ir al médico, ir a nuestro centro de estética, reservar hora para el taller etc.

Esta aplicación permite al gestor de un centro fidelizar a sus clientes, no tiene que atender llamadas improductivas y puede publicar promociones especiales para clientes de la aplicación.

El consumidor, por su parte, puede realizar reservas de una forma clara y fácil con la antelación que desee, además de aprovechar promociones exclusivas por ser usuario de la aplicación.

Funcionalidad

En la aplicación podrán acceder dos tipos diferentes de usuarios, por un lado podemos entrar como empresa que hace uso de la aplicación para gestionar la agenda donde están hechas las reservas, y por otro lado podemos entrar como usuario que desea realizar una reserva.

Funcionalidad para la empresa

La empresa podrá consultar su agenda personal para ver las reservas realizadas por los usuarios y podrá reservar también hora para los usuarios, es importante que siempre se use la aplicación para reservar hora para que no se genere duplicidad de reservas.

De esa manera la aplicación será su herramienta para gestionar las reservas que sustituirá a su libreta o agenda habitual.

Funcionalidad para el usuario

- Podrá reservar fecha y hora para un evento y seleccionar opciones del evento, por ejemplo podrá pedir hora para cortarse el pelo o teñírselo la aplicación tendrá en cuenta el número de profesionales disponibles para poder realizar la reserva en una franja horaria, si el servicio tiene 2 profesionales se podrán realizar 2 reservas en la misma franja horaria.

- Podrá configurar un recordatorio periódico para la reserva de eventos, por ejemplo si nos cortamos el pelo cada 4 semanas aproximadamente la aplicación le recordará al usuario si desea realizar la reserva cuando pasen 3 semanas para reservarle hora automáticamente para la 4ª semana.

- Podrá buscar que empresas disponen de la aplicación para poder realizar sus reservas o realizar una búsqueda por el nombre de la empresa.

- Podrá consultar en un mapa cerca de su ubicación que tipo de empresas registradas tiene cerca o en un lugar específico.

Funcionamiento de la aplicación

La aplicación funcionará sincronizando la agenda de cada empresa en el momento en el que se acceda a dicha agenda, será necesario que los usuarios se registren una vez se instale la aplicación para poder utilizarla.

En caso de ser una empresa el distribuidor de la aplicación será el encargado de darla de alta en el servicio y configurar el tipo de empresa y los servicios que ofrece.

1.3 Requisitos

Funcionales:

- Dispositivo con sistema operativo Android y acceso a internet.

Técnicos:

- Herramientas case apropiadas para el desarrollo de la aplicación, drivers, máquinas virtuales, kits de desarrollo etc.

- Una base de datos donde se guardan los datos de las empresas que estén dadas de alta, y los datos de los usuarios registrados.

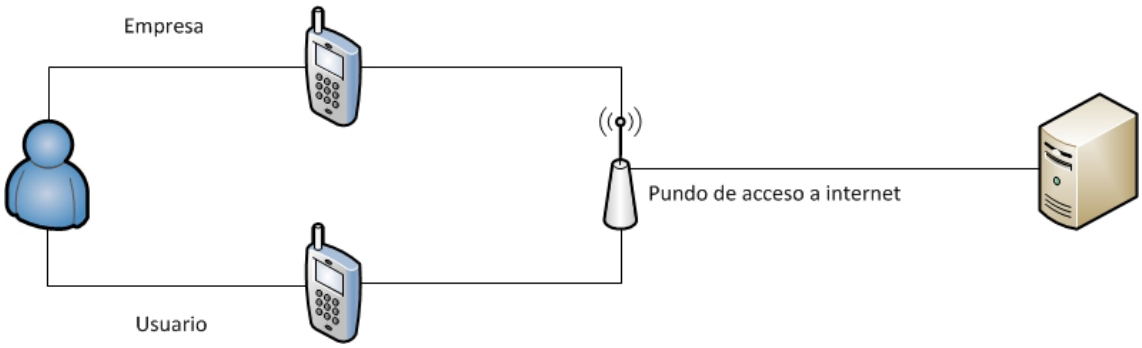


Ilustración 1. Arquitectura física

1.4 Formación previa al entorno

Al tratarse de un proyecto donde se utilizarán nuevas tecnologías antes de poder realizar las fases iniciales del proyecto será conveniente tener una formación básica de dichas tecnologías para afrontar mejor el análisis.

Objetivo:

El objetivo es conocer los kits de desarrollo (SDK) para Android, como lanzar una aplicación piloto y debugar sobre un dispositivo físico, conocer las arquitecturas y el esqueleto básico de una aplicación desarrollada para un sistema operativo Android así como las Apis más importantes y comunes para el desarrollo de la aplicación, y conocer cómo trabaja a nivel de archivos, datos, accesos, hardware etc.

Realizar un pequeño estudio de información que podemos encontrar tanto en internet como en libros que nos puedan servir de ayuda y apoyo durarte la realización del proyecto.

1.5 Fases

Para poder seguir un orden en la elaboración del proyecto seguiremos las siguientes fases:

Análisis Previo, Análisis (Planificación y especificación), Diseño, Programación (Implementación), Prueba (Test) y documentación final.

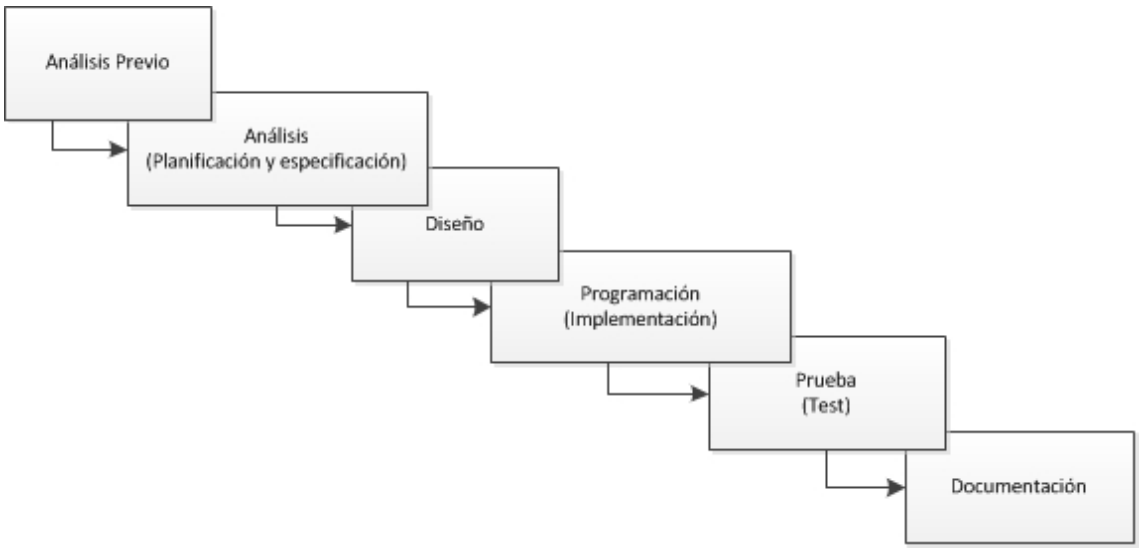


Ilustración 2. Fases

1.5.1 Análisis Previo

Objetivo:

Los objetivos de esta fase son definir la idea y el propósito de la aplicación con el fin de tener un punto claro de partida sobre el trabajo que se va a realizar y sobre que sistemas y quien lo va a utilizar.

Documentación:

La documentación de esta fase constará de un resumen donde se explicará la idea básica de la aplicación y los requisitos mínimos para el desarrollo del proyecto y las personas que participarán.

Finalidad y riesgos:

Este resumen nos tiene que servir para poder consultar cual es la idea del proyecto a grandes rasgos y tener un punto de partida sobre lo que vamos a trabajar.

1.5.2 Análisis

Objetivo:

El objetivo de esta fase es definir con detalle la funcionalidad de la aplicación, esta tiene 2 puntos claros de funcionalidad que son la parte de empresa y cliente en esta fase especificaremos detalladamente la funcionalidad de cada perfil, tendremos que realizar una especificación clara de la funcionalidad detallando cada caso de uso con esquemas teniendo en cuenta las personas que utilizarán la aplicación y el propósito de la misma.

También definiremos las pantallas sobre las que interactuarán los usuarios de la aplicación, para tener una visión clara de la función de cada pantalla.

Documentación:

Resumen detallado de la funcionalidad.

Diagramas de casos de uso de los usuarios.

Imágenes donde se mostrará el diseño de las pantallas con los controles correspondientes para su utilización tanto para los casos de empresa como de cliente.

Finalidad y riesgos:

Esta documentación nos servirá para enfocar la fase de diseño y la fase de implantación teniendo en cuenta la explicación de la funcionalidad, casos de uso y pantallas funcionales.

1.5.3 Diseño

Objetivo:

Definir a partir de los datos de la fase de análisis la arquitectura que tendrá la aplicación para poder resolver la funcionalidad.

Tenemos que especificar como resolveremos funcionalmente los siguientes puntos:

Uso de agendas sincronizadas.

Accesos a bases de datos para altas, bajas, modificaciones, consultas.

Configuración de cuentas para inicios de sesión.

Utilización de Apis para la utilización de GPS y posicionamiento.

Definición de tipo de controles utilizados para mostrar correctamente la información y que el uso funcional de la aplicación sea intuitivo.

Otro de los puntos que hay que definir en esta fase es realizar un estudio sobre Apis que nos ofrecen almacenamiento con tecnología (the cloud) en la nube, para ver si es una buena opción para almacenar información.

Documentos:

Documentó de texto donde se especificará las diferentes funciones más importantes de la aplicación.

Escenarios de uso.

Resumen de personajes.

Diagramas de flujo.

Prototipos de pantallas.

Finalidad y riesgos:

Los datos recogidos en esta fase y los documentos elaborados nos servirán para realizar la fase de programación y poder detectar complicaciones que nos retrasen en el tiempo y evaluar la complejidad de algunas funcionalidades.

1.5.4 Programación

Objetivo:

El objetivo de esta fase es implementar el código la especificación y arquitectura definida en la fase de diseño.

En esta fase montaremos los diferentes módulos o clases que utilizaremos para la realización de la aplicación, programaremos sus funciones internas, y pondremos en práctica el uso de las Apis aprendidas durante la fase de formación para los diferentes puntos de la aplicación como por ejemplo el uso de localización GPS, los accesos a datos, control del hardware del dispositivo etc.

Se diseñará la parte de interface gráfica para utilizar gráficos que sean intuitivos en la aplicación y que tengan resoluciones gráficas que sean fáciles de ajustar a diferentes tipos de dispositivos.

Documentos:

Archivos necesarios para realizar una instalación.

Documentó de arquitectura.

Código fuente.

Manual de instalación

Finalidad y riesgos:

Durante el desarrollo de esta fase tendremos que contemplar que los tiempos pueden ser muy variables según la dificultad que nos encontremos a la hora de utilizar Apis, solucionar problemas que no se vieron en el diseño, o problemas típicos a la hora de programar, por eso hay que evaluar bien los tiempos en esta fase y contemplar que se puedan realizar cambios sobre la marcha que agilicen la entrega del producto final.

1.5.6 Prueba Test

Objetivo:

En esta fase realizaremos pruebas de test para asegurar el correcto funcionamiento de la aplicación, se realizarán pruebas de test sobre las funcionalidades más relevantes.

Acceso a bases de datos.

Gestión de Reservas.

Muestreo de información.

Pruebas de GPS.

Se realizarán pruebas por separado para la parte de cliente y empresa.

Finalidad y riesgos:

Hay que contemplar el tiempo para la corrección de errores encontrados y las siguientes pruebas para confirmar que estos errores se han solucionado.

Documentos:

Resumen de banco de pruebas realizado.

Resumen de errores encontrados y corregidos.

1.5.7 Documentación

Objetivo:

El objetivo de esta fase es desarrollar la memoria del proyecto junto con un manual de usabilidad de la aplicación.

Documentos:

Memoria del proyecto.

Manual de usabilidad.

1.6 Diagrama de Gantt

Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
Formación previa al entorno	4 días	mar 02/10/12	vie 05/10/12	
Análisis Previo	2 días	lun 08/10/12	mar 09/10/12	1
Análisis	5 días	mié 10/10/12	mar 16/10/12	2
Diseño	8 días	mié 10/10/12	vie 19/10/12	2
Programación (Implantación)	25 días	lun 22/10/12	vie 23/11/12	4
Pruebas de Test	11 días	lun 26/11/12	lun 10/12/12	5
Documentación (Memoria)	20 días	mar 11/12/12	lun 07/01/13	6

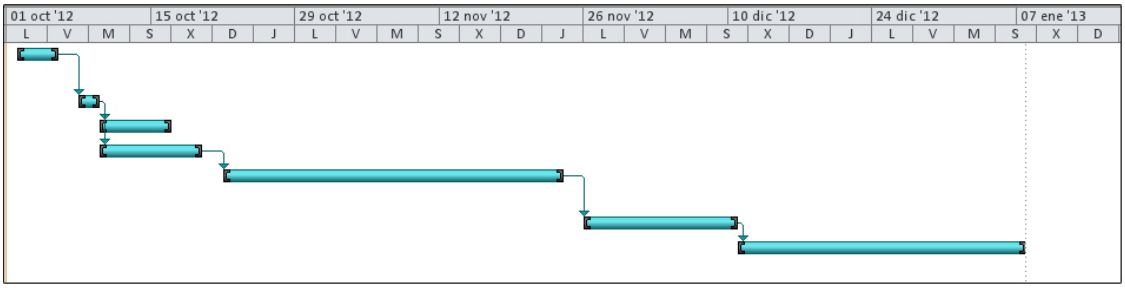


Ilustración 3. Diagrama de Gantt, fases de trabajo

2. Diseño centrado en el usuario

2.1 Introducción

En este apartado se integrará el plan de trabajo en el modelo de diseño centrado en el usuario (DCU) definiendo cuatro fases claves:

- Usuarios y contexto.
- Diseño conceptual.
- Prototipo.
- Evaluación.

Observaciones:

En el primer apartado donde definíamos el guión de trabajo se diseñó un guión con modelo en cascada combinado con diseño centrado en el usuario, después de leer y estudiar el diseño centrado en el usuario se sacan como conclusiones que el modelo en cascada no es el más apropiado para desarrollos sobre dispositivos móviles de ese modo en este apartado se integrará el modelo de trabajo del apartado de guía de trabajo con las fases estudiadas en el diseño centrado en el usuario.

2.2 Objetivo:

El objetivo de este apartado es poder tener documentado el proyecto con un diseño basado en la usabilidad y la simplicidad de cara al uso del usuario final en la aplicación, este documento nos servirá para desarrollar la fase de programación de la aplicación.

2.3 Usuarios y Contexto de usos.

Objetivo:

El objetivo de esta fase es involucrar a los usuarios finales de la aplicación para conocer el perfil de los usuarios, obtener información sobre sus necesidades, su manera de interactuar con la aplicación y las funcionalidades que esta tendría que tener utilizando los métodos oportunos.

2.3.1 Métodos:

Para realizar esta fase y conocer los perfiles de usuarios y su contexto de uso utilizaremos los siguientes métodos:

- Observación e investigación contextual.
- Entrevistas en profundidad.
- Análisis competitivo.

Se ha decidido hacer entrevistas en profundidad porque se tenía la posibilidad de evaluar personalmente a varios usuarios finales de la aplicación para conocer sus necesidades y ver el tipo de perfiles.

Hay algunas aplicaciones de reservas para otros ámbitos como restaurantes que nos podían servir de gran ayuda a la hora de coger ideas para tener una funcionalidad cercana a lo que los usuarios necesitan por eso también se realiza un análisis competitivo.

Perfiles de usuarios

Para el uso de la aplicación sobre la que trabajamos en este proyecto hay dos perfiles de usuarios o personajes, por un lado el perfil de empresa y por otro el de cliente.

Perfiles de empresa:

Este perfil de usuario es un propietario de una pequeña o mediana empresa.

Es un perfil con edades comprendidas entre 25 – 60 años de edad.

El perfil puede tener un rango de experiencia en el uso de tecnologías móviles muy abierto, desde un buen conocimiento de la tecnología hasta un usuario con muy poca experiencia.

La motivación principal del tipo de perfil es el crecimiento de su negocio, así como mantener una buena reputación y fidelización con los clientes.

Las necesidades de estos clientes son:

- Control de las reservas.
- Gestión de la agenda, poder realizar reservas manuales.
- Flexibilidad en las reservas.
- Recordatorios para clientes en futuras reservas.
- Posicionamiento de su negocio.
- Fidelización con los clientes.

Observaciones mostradas por este tipo de perfil en entrevista personal:

Existe la posibilidad que usuarios del perfil cliente realicen una reserva y después no se presenten a la cita, esto puede suponer gastos al negocio y es interesante tener un control sobre este tipo de acciones, como ejemplo proponen sancionar al usuario vetando el uso a la aplicación si realiza más de 3 reservas y sin cancelarlas después no acude a la cita.

Otro dato a tener en cuenta es que los usuarios comprendidos entre edades más jóvenes suelen no presentarse a citas sobre reservas hechas para festivos o fines de semana por la mañana.

Perfiles de clientes:

El perfil de cliente es un usuario que dispone de un dispositivo móvil con tecnología Android y tarifa de datos para poder realizar reservas sobre los servicios.

Serán usuarios comprendidos entre los 18 – 60 años de edad.

Generalmente el perfil de usuario que utiliza la aplicación tiene una experiencia media en el uso de aplicaciones táctiles sobre dispositivos móviles.

Sus motivaciones son poder tener un centro o servicio donde ir habitualmente y el hecho de ser un cliente asiduo nos permite que se le pueda recompensar con ventajas sobre ese servicio.

Las necesidades de estos clientes son:

- Poder realizar reservas en cualquier franja horaria y día.
- Saber cuándo hay fechas disponibles para una reserva.
- Encontrar servicios cerca de su vivienda o lugar actual donde se encuentre.
- Tener un recordatorio de reservas.
- Tener una indicación del lugar de la cita para nuevos servicios.

Observaciones mostradas por este tipo de perfil en entrevista personal:

Poder cancelar las reservas, poder tener una vista rápida de las horas y días disponibles para poder ver la reserva, si se realizan x reservas seguidas a un mismo servicio que estas queden registradas para que el usuario del perfil empresa pueda tener una compensación con el cliente en modo de descuentos ofertas etc.

Contexto de uso

Contexto en perfil de empresa:

El escenario donde los usuarios con este perfil normalmente usarán la aplicación será en su lugar de trabajo o negocio con acceso a internet, los horarios generalmente serán en función a su jornada laboral y ocasionalmente fuera del horario de trabajo y en escenarios donde tenga acceso a internet.

Contexto en perfil de cliente:

El escenario de este perfil de usuario puede ser cualquier lugar con acceso a internet desde donde el usuario pueda realizar con comodidad la reserva, en su casa, viajando en transporte públicos, esperando a un acontecimiento etc.

El horario podrá ser en cualquier momento del día generalmente horario de tarde después de la jornada laboral.

2.3.2 Análisis de tareas

En este apartado se va a profundizar más en las necesidades y tareas que los usuarios realizarán cuando utilicen la aplicación.

Análisis perfil de empresa:

El usuario que utiliza la aplicación como empresa necesitará realizar las siguientes tareas:

Control de reservas: Podrá consultar su agenda personal sobre un calendario verá las reservas que hay hechas y podrá consultarlas detalladamente.

Gestión de la agenda: En su agenda personal podrá realizar reservas que los usuarios le pidan personalmente o modificar o eliminar otras reservas.

Editar su cuenta: Podrá editar los datos personales de su negocio y de su cuenta de usuario.

Análisis perfil de cliente:

Gestión de su cuenta: El cliente se puede dar de alta en la aplicación y posteriormente eliminar o modificar su cuenta.

Realización de reservas: Podrá realizar reservas sobre los servicios que necesite en cualquier franja horaria y cualquier día.

Consulta de reservas: Podrá tener una visión del calendario para poder ver si hay reservas disponibles sobre un servicio de manera ágil.

Búsquedas: El cliente puede realizar búsquedas para los diferentes servicios donde quiera realizar una reserva, las búsquedas se pueden realizar a través del nombre del servicio, por localización GPS o sobre una lista de servicios.

Recordatorio: Se podrá editar un recordatorio para realizar reservas, por ejemplo que avise cada cuatro semanas para reactivar la reserva para ir a cortarse el pelo.

Localización: Cuando se realicen reservas se podrá consultar en un mapa la localización GPS del servicio al que tenemos que acudir, tendremos un enlace en la reserva para poder ver el mapa.

Reservas realizadas: Se tendrá un listado donde se podrán ver todas las reservas realizadas que tenemos pendientes y se podrán consultar detalladamente.

2.3.3 Análisis competitivo

Se han analizado las siguientes aplicaciones para apoyarnos en el diseño que utiliza la competencia con servicios similares al que desarrollara nuestra aplicación.

Aplicación	Ideas
El Tenedor. Aplicación de reservas para restaurantes.	Ha aportado ideas de diseño y estructura de pantallas.
Réstalo. Aplicación de reservas para restaurantes.	Ha aportado ideas de usabilidad, filtros de búsqueda y diseño.
Tripadvisor. Aplicación de búsqueda de servicios.	Ha aportado ideas de localización geográfica.
Bookitit. Aplicación de gestión de agendas.	Ha aportado ideas para registro de usuarios y diseños de pantallas de agenda.

2.4 Diseño Conceptual.

En esta fase veremos el diseño conceptual del proyecto basándonos en los escenarios de uso y en la estructura de la aplicación.

2.4.1 Planteamiento de integración sobre el DCU

Para llevar a cabo la parte de diseño conceptual se ha colaborado con varios actores o personajes de muestra uno por parte de empresa y otro por la parte de cliente, podemos consultar los resultados de los test o encuetas en el punto 2.6.2 Puesta en práctica.

Planteamiento empresa

Empresa, peluquería la fuente.	
Escenario	Centro de peluquería situado en Badalona.
Estudio	Entrevista personal, encuesta.
Contexto	Día normal en el que me dirijo a realizar un servicio y seguidamente se aprovecha para realizar el estudio y hacer un análisis de la clientela.
Justificación	Es una empresa que está interesada en el servicio y podía ofrecer información sobre las necesidades y las funciones que debería tener la aplicación.
Resultados	Tanto la encuesta como la entrevista personal dieron un resultado satisfactorio que resolvió muchas dudas y dio a conocer aspectos importantes para el cliente que se desconocían para tener en cuenta sobre todo en el uso de la aplicación por parte de los usuarios.

Empresa, taller AutoMarly.	
Escenario	Taller situado en el barrio de Sant Andreu, Barcelona.
Estudio	Entrevista personal, encuesta.
Contexto	Al finalizar el día laboral en las oficinas del taller se realiza una entrevista personal y una encuesta al propietario.
Justificación	Responsable de un negocio que sería un potencial cliente para realizar reservas y que estaría dispuesto a probar una beta funcional del producto.
Resultados	Resultado satisfactorio en la entrevista, la encuesta ha servido para pensar en un desarrollo con una interface fácil de utilizar para cualquier usuario sobre todo de cara a la empresa y con las opciones básicas para la realización de reservas que no complique el uso de la aplicación.

Planteamiento cliente

Clientes de servicio concreto.	
Escenario	Centro de peluquería situado en Badalona.
Estudio	Encuesta.
Contexto	Durante la realización del servicio se aprovecha para realizar una encuesta al cliente.
Justificación	Se aprovecha la visita al centro para realizar encuestas también a sus clientes.
Resultados	Los clientes se muestran interesados y la entrevista muestra un perfil propenso a la utilización de la aplicación, hay usuarios con más facilidad en el uso de nuevas tecnologías que otros.

Clientes de servicio general.	
Escenario	Hogar del cliente.
Estudio	Entrevista personal, encuesta.
Contexto	Después de la jornada laboral del cliente una vez en su hogar se le realiza una entrevista y una encuesta.
Justificación	Persona cercana que se prestaría a probar la aplicación y con tiempo para realizar la entrevista personal y la encuesta.
Resultados	El cliente realiza una crítica constructiva dando nuevas ideas y aportes que considera le darían un valor añadido la aplicación y valora positivamente muchos aspectos de la misma.

2.4.2 Personajes

En este apartado se muestran 4 ejemplos de personajes que podrían utilizar la aplicación unos en el perfil de empresa y otros en el perfil de cliente, estos personajes se elaboran con la información extraída de los test de usuario del apartado de evaluación y la entrevista personal.

Perfil Empresa

<table>
<tr><td>  </td><td>Persona:

Alberto es el dueño de una peluquería, tiene 32 años de edad, vive cerca de su negocio y pasa la mayor parte del día en la peluquería trabajando. Suele salir una hora al medio día para comer y a las 8 de la tarde termina su jornada laboral, una vez termina se desentiende totalmente del trabajo y se dedica a otras actividades como salir a correr.

Trabaja los sábados y tiene como día libre el lunes que hay menos clientes.</td></tr>
<tr><td colspan="2">Escenario:

Pequeña peluquería situada en Badalona en el barrio de Lloreda, en una calle transitada y rodeada por otros pequeños negocios.</td></tr>
<tr><td colspan="2">Relación con la tecnología:

Suele utilizar su Smartphone para conectarse a internet, consultar noticias y su correo, se conecta habitualmente en el trabajo en momentos puntuales o de descanso y también una vez terminada su jornada laboral.

En casa a pesar de tener ordenador con conexión a internet no suele utilizarlo.</td></tr>
<tr><td colspan="2">Objetivo:

Alberto tiene clientes que son asiduos a la peluquería desde hace muchos años, a Alberto le gustaría poder premiar a estos clientes controlando las veces que han acudido a cortarse el pelo y dándoles también una opción más cómoda para realizar reservas en horarios que la peluquería no está abierta.</td></tr>
<tr><td colspan="2">Interacción:

Es una persona que está acostumbrada al uso de aplicaciones sencillas para todo el público con una interface de usuario clara y sencilla que le permita intuir con facilidad los pasos a seguir para realizar una tarea específica dentro de la aplicación.</td></tr>
</table>

Persona:

José Antonio es el dueño de un taller, tiene 52 años, entra en el taller a las 8 de la mañana y realiza un descanso de 2 horas para comer normalmente entre las 2 y las 4 de la tarde, su jornada laboral no termina hasta las 9 o 10.

Una vez terminada su jornada laboral va a casa, se suele acostar pronto, el fin de semana no abre y aprovecha para descansar y estar con su familia.

Escenario:

Taller de tamaño mediano en un polígono situado en Barcelona en el barrio de Sant Andreu, hay más negocios similares a su alrededor, no es una zona de mucho tránsito.

Relación con la tecnología:

Suele utilizar un pequeño ordenador en la oficina del taller para realizar gestiones y conectarse a internet.

Utiliza su Smartphone para realizar llamadas y utilizar alguna aplicación de utilidad como agendas o blocs de notas.

Objetivo:

José Antonio quiere que sus clientes lo localicen cuando busquen talleres cercanos y poder tener un control de las reservas en contenido digital para poder consultarlas en cualquier momento.

Interacción:

Es una persona que está acostumbrada al uso de aplicaciones de gestión, está acostumbrada a moverse por menús , enviar mails , introducir datos etc.

Perfil cliente

Persona:

Javi tiene 32 años, está estudiando inglés y terminando la carrera de Ingeniería Técnica, suele ocupar las mañanas y parte de la tarde para estudiar.

En su tiempo libre sale a correr o cuida de sus acuarios.

Escenario:

Piso donde vive en Barcelona, barrio de la Zona franca, está rodeado de más viviendas y tiene pocos servicios cerca de casa.

Relación con la tecnología:

Siempre está en contacto con la tecnología ya sea a través de su ordenador personal su portátil o su Smartphone, lo utiliza principalmente para estudiar, mantenerse informado o entrar en redes sociales, de vez en cuando utiliza alguna aplicación de diseño grafico o juega algún videojuego.

Tiene acceso a internet en cualquier franja horaria y situación.

Objetivo:

Javi es una persona bastante fiel a los servicios que suele utilizar, lleva 15 años acudiendo a la misma peluquería, le gustaría que esta fidelidad se viera premiada, cree que una buena opción sería que sus visitas quedaran reflejadas.

Interacción:

Es una persona que está acostumbrada al uso de cualquier tipo de aplicaciones, aunque le gusta que las aplicaciones tengan un entorno atractivo y fácil de utilizar.

Persona:

Raquel tiene 34 años, trabaja por las tardes en un centro de servicios, trabaja 6 horas de 4 de la tarde a 10 de la noche.

Le gusta jugar al póker, ver películas y series de televisión e ir en bicicleta.

Dedica las mañanas a estudiar o salir en bicicleta.

Escenario:

Piso donde vive en Barcelona, barrio de la Zona franca, está rodeado de más viviendas y tiene pocos servicios cerca de casa.

Relación con la tecnología:

Utiliza un portátil para estudiar, jugar y ver contenidos multimedia, también utiliza un Smartphone para mantenerse en contacto con familiares y amigos y consultar su correo.

Tiene acceso a internet solo cuando está en casa.

Objetivo:

Raquel acude al dentista cada 6 meses para realizar una limpieza bucal, le gustaría poder tener un recordatorio para que la aplicación le recuerde que tiene que ir a realizarse la limpieza y de forma rápida pueda realizar la reserva.

Interacción:

Es una persona que está acostumbrada al uso de aplicaciones de gestión, videojuegos y redes sociales, no le gusta perder mucho tiempo aprendiendo a utilizar una aplicación, aunque es bastante autodidacta y suele investigar para ver cuál es su funcionamiento.

2.4.3 Escenarios de uso

Perfil de usuario empresa:

Escenario de uso consultar reservas en perfil empresa.

Escenario de uso consultar reservas.	
Usuario	Empresa
Lugar o escenario	Centro de trabajo
Objetivo	Consultar lista de reservas pendientes
Tareas	Entrar en el menú agenda en la aplicación, y seleccionar el día del que se quieren visualizar las reservas, salir de la agenda.
Información necesaria	Necesita saber para qué día quieres consultar las reservas.
Funcionalidades	Sincronización del calendario, acceso al calendario, visualización de datos de las reservas en lista.

Flujo principal

1. El usuario entra en la aplicación como empresa.
2. La aplicación sincroniza los datos con el servidor.
3. El usuario entra en el menú de agenda.
4. La aplicación muestra la agenda en pantalla.
5. El usuario selecciona el día que quiere consultar.
6. La aplicación muestra una lista con las reservas para ese día en orden horario.
7. El usuario visualiza en pantalla las reservas.
8. El usuario sale de la pantalla de reservas.

Escenario realizar reserva en perfil empresa.

Escenario de uso realizar reservas.	
Usuario	Empresa
Lugar o escenario	Centro de trabajo
Objetivo	Realizar una reserva
Tareas	Entrar en el menú reservas en la aplicación, y seleccionar el día y la hora para el que se quieren realizar las reservas

Información necesaria	Necesita saber para qué día y para quien es la reserva.
Funcionalidades	Sincronización del calendario, acceso al calendario, edición de datos para la reserva.

Flujo principal

1. El usuario entra en la aplicación como empresa.
2. La aplicación sincroniza los datos con el servidor.
3. El usuario entra en el menú de reserva.
4. La aplicación muestra la agenda en pantalla.
5. El usuario selecciona el día que quiere realizar la reserva.
6. La aplicación muestra una pantalla para editar la hora de la reserva.
7. El usuario edita la reserva y la guarda.
8. La aplicación acepta la reserva y sincroniza la información.
9. El usuario sale de la pantalla de edición de reservas.

Flujos alternativos

- Ya hay una reserva para esa hora, la aplicación no acepta la reserva e informa al usuario.

Escenario editar reservas en perfil empresa.

Escenario de uso editar reservas.	
Usuario	Empresa
Lugar o escenario	Centro de trabajo
Objetivo	Editar una reserva
Tareas	Entrar en el menú reservas en la aplicación, y seleccionar el día y la hora para el que se quieren editar la reserva.
Información necesaria	Necesita saber para qué día se quiere editar la reserva.
Funcionalidades	Sincronización del calendario, acceso al calendario, visualización de datos de las reservas en lista, edición de datos para la reserva.

Flujo principal

1. El usuario entra en la aplicación como empresa.
2. La aplicación sincroniza los datos con el servidor.

3. El usuario entra en el menú de reserva.

4. La aplicación muestra la agenda en pantalla.

5. El usuario selecciona el día que quiere editar la reserva.

6. La aplicación muestra una pantalla la lista de reservas de ese día.

7. El usuario selecciona la reserva que quiere editar.

8. La aplicación muestra una pantalla con los datos de la reserva.

9. El usuario modifica o elimina la reserva.

10. La aplicación edita y sincroniza la información.

11. El usuario sale de la pantalla de edición de reservas.

Flujos alternativos

- La aplicación no puede editar la información porque ya hay una reserva para esa hora.

Escenario de edición de cuenta en perfil empresa.

Escenario de uso editar cuenta.	
Usuario	Empresa
Lugar o escenario	Centro de trabajo
Objetivo	Editar su cuenta
Tareas	Entrar en el menú de cuenta de la aplicación y editar la cuenta.
Información necesaria	Datos que quiere editar de la cuenta.
Funcionalidades	Sincronización de cuenta, acceso a la cuenta de usuario, visualización de datos de la cuenta, edición de datos para la cuenta de usuario.

Flujo principal

1. El usuario entra en la aplicación como empresa.

2. La aplicación sincroniza los datos con el servidor.

3. El usuario entra en el menú de cuenta.

4. La aplicación muestra los datos de su cuenta en pantalla.

5. El usuario selecciona el control para edición de cuenta.

6. La aplicación muestra en pantalla los datos editables de la cuenta.

7. El usuario edita los valores de la cuenta.

8. La aplicación edita y sincroniza la información.

9. El usuario sale de la pantalla de edición de cuenta.

10. El usuario sale de la aplicación.

Flujos alternativos

- La aplicación no puede editar la información porque contiene errores.

Perfil de usuario cliente:

Escenario de registro de usuario en perfil cliente.

Escenario de uso registro de usuario.	
Usuario	Cliente
Lugar o escenario	Hogar del usuario.
Objetivo	Registrarse en la aplicación como usuario
Tareas	Entrar en la aplicación y registrar sus datos de usuario
Información necesaria	Datos personales para dar de alta, nombre de usuario, dirección de correo electrónico y contraseña.
Funcionalidades	Registro de datos.

Flujo principal

1. El usuario entra en la aplicación como nuevo cliente.

2. La aplicación muestra la pantalla de registro.

3. El usuario introduce sus datos para el registro.

4. La aplicación guarda los datos y registra al usuario.

5. La aplicación entra en el menú principal.

5. El usuario sale de la aplicación.

Flujos alternativos

- La aplicación no puede registrar al usuario porque ya existe o los datos no son válidos y vuelve a la pantalla de registro.

Escenario de consulta de reservas en perfil cliente.

Escenario de consulta de reservas.	
Usuario	Cliente
Lugar o escenario	Hogar del usuario.
Objetivo	Consultar sus reservas
Tareas	Entrar en la aplicación y ver las reservas que tiene hechas
Información necesaria	Fecha y hora de las reservas que se quieren consultar.
Funcionalidades	Sincronización del calendario, acceso al calendario, visualización de datos de las reservas en lista.

Flujo principal

1. El usuario entra en la aplicación como cliente.
2. El usuario entra en el menú de calendario.
3. El usuario pulsa el día que quiere consultar las reservas.
4. La aplicación muestra una lista con las reservas hechas.
5. El usuario pulsa la reserva que quiere consultar detalladamente.
6. La aplicación muestra los datos de la reserva.
7. El usuario sale de la aplicación.

Flujos alternativos

- La aplicación avisa de que no hay reservas y vuelve a la pantalla del calendario.

Escenario de realizar reservas en perfil cliente.

Escenario de realizar reservas cliente.	
Usuario	Cliente
Lugar o escenario	Hogar del usuario.
Objetivo	Realizar una reserva.
Tareas	Entrar en la aplicación y realizar una reserva para el día y la hora deseado.
Información necesaria	Fecha y hora de las reservas que se quieren hacer.
Funcionalidades	Sincronización del calendario, búsqueda con filtro, edición de datos de las reservas.

Flujo principal

1. El usuario entra en la aplicación como cliente.

2. El usuario entra en el menú de búsqueda de servicio.

3. El usuario busca un servicio.

4. La aplicación muestra una lista con los servicios que se corresponden con el filtro de búsqueda.

5. El usuario selecciona el servicio.

6. La aplicación muestra los datos del servicio.

7. El usuario pulsa el control para realizar una reserva.

8. La aplicación muestra la pantalla para introducir los datos de la reserva.

9. El usuario introduce los datos para la reserva (fecha y hora).

10. La aplicación guarda los datos y realiza la reserva.

11. El usuario sale de la aplicación.

Flujos alternativos

- La aplicación no puede realizar la reserva porque ya hay una reserva para esa franja horaria e informa al usuario.

Escenario de editar reservas en perfil cliente.

Escenario de editar reservas.	
Usuario	Cliente
Lugar o escenario	Hogar del usuario.
Objetivo	Editar una reserva.
Tareas	Entrar en la aplicación y editar una reserva creada con anterioridad, eliminarla o modificarla.
Información necesaria	Fecha y hora de las reservas que se quieren editar.
Funcionalidades	Sincronización del calendario, edición de datos de las reservas.

Flujo principal

1. El usuario entra en la aplicación como cliente.

2. El usuario entra en el menú de reservas.

3. La aplicación muestra una lista con sus reservas.

4. El usuario selecciona la reserva que quiere editar.

5. El usuario elimina o modifica la reserva.

6. La aplicación elimina o modifica la reserva.

7. El usuario sale de la aplicación.

Flujos alternativos

- La aplicación no puede modificar la reserva porque la fecha no está disponible o los datos no son correctos.

- La aplicación ya no deja modificar o eliminar la reserva porque se tiene que hacer con más de una hora de antelación.

Escenario de realizar búsquedas en perfil cliente.

Escenario de realizar búsquedas.	
Usuario	Cliente
Lugar o escenario	Hogar del usuario.
Objetivo	Realizar una búsqueda.
Tareas	Entrar en la aplicación y realizar una búsqueda con uno de los filtros de búsqueda.
Información necesaria	Nombre del Servició o Localización.
Funcionalidades	Sincronización de aplicación, consultas a base de datos, búsqueda localización geográfica.

Flujo principal

1. El usuario entra en la aplicación como cliente.

2. El usuario entra en el menú de búsquedas.

3. El usuario selecciona el tipo de servicio que quiere buscar.

4. La aplicación muestra una pantalla con las opciones de búsqueda.

5. El usuario selecciona buscar según la opción o filtro de búsqueda que más le interese.

6. La aplicación muestra una lista con los servicios.

7. El usuario selecciona el servicio para consultar sus datos detalladamente.

8. El usuario sale de la aplicación.

Flujos alternativo

- En la búsqueda por nombre la aplicación busca el servicio si lo encuentra lo muestra en pantalla si no da un aviso.

- En la búsqueda por localización geográfica la aplicación muestra un mapa con los servicios cerca de la localización indicada o cerca de la posición GPS del usuario.

Escenario de realizar un recordatorio en perfil cliente.

Escenario de realizar un recordatorio.	
Usuario	Cliente
Lugar o escenario	Hogar del usuario.
Objetivo	Realizar un recordatorio o alarma.
Tareas	Entrar en la aplicación y realizar recordatorio para próximas reservas.
Información necesaria	Tipo de servicio y tiempo para el recordatorio.
Funcionalidades	Guardar datos del recordatorio, y activar servicio de recordatorio.

Flujo principal

1. El usuario entra en la aplicación como cliente.

2. El usuario entra en el menú de reservas.

3. La aplicación muestra una lista con las reservas.

4. El usuario selecciona una reserva.

5. La aplicación muestra los datos de la reserva.

6. El usuario pulsa el control para realizar un recordatorio.

7. La aplicación muestra una pantalla para editar los datos del recordatorio.

8. El usuario configura su recordatorio para ese servicio.

9. La aplicación guarda los datos y activa el servicio de aviso de recordatorio para ese usuario.

10. El usuario sale de la aplicación.

Flujos alternativo

- Los datos del recordatorio no son correctos la aplicación avisa al usuario y vuelve a la pantalla donde se muestra la reserva.

Escenario de uso editar cuenta en perfil cliente.

Escenario de uso editar cuenta.	
Usuario	Cliente
Lugar o escenario	Hogar del usuario.
Objetivo	Editar su cuenta
Tareas	Entrar en el menú de cuenta de la aplicación y editar la cuenta.
Información necesaria	Datos que quiere editar de la cuenta.
Funcionalidades	Sincronización de cuenta, acceso a la cuenta de usuario, visualización de datos de la cuenta, edición de datos para la cuenta de usuario.

Flujo principal

1. El usuario entra en la aplicación como cliente.
2. La aplicación sincroniza los datos con el servidor.
3. El usuario entra en el menú de cuenta.
4. La aplicación muestra los datos de su cuenta en pantalla.
5. El usuario selecciona el control para edición de cuenta.
6. La aplicación muestra en pantalla los datos editables de la cuenta.
7. El usuario modifica los valores de la cuenta o elimina la cuenta.
8. La aplicación modifica o elimina la cuenta y sincroniza la información.
9. El usuario sale de la pantalla de edición de cuenta.
10. El usuario sale de la aplicación.

Flujos alternativos

- La aplicación no puede modificar la cuenta porque la información introducida contiene errores.

2.4.4 Flujos de interacción

Perfil de usuario empresa

Diagrama de flujo consultar reservas.

Este diagrama nos muestra la estructura en el flujo de desarrollo cuando un usuario de tipo empresa quiere consultar el estado o los detalles de una reserva.

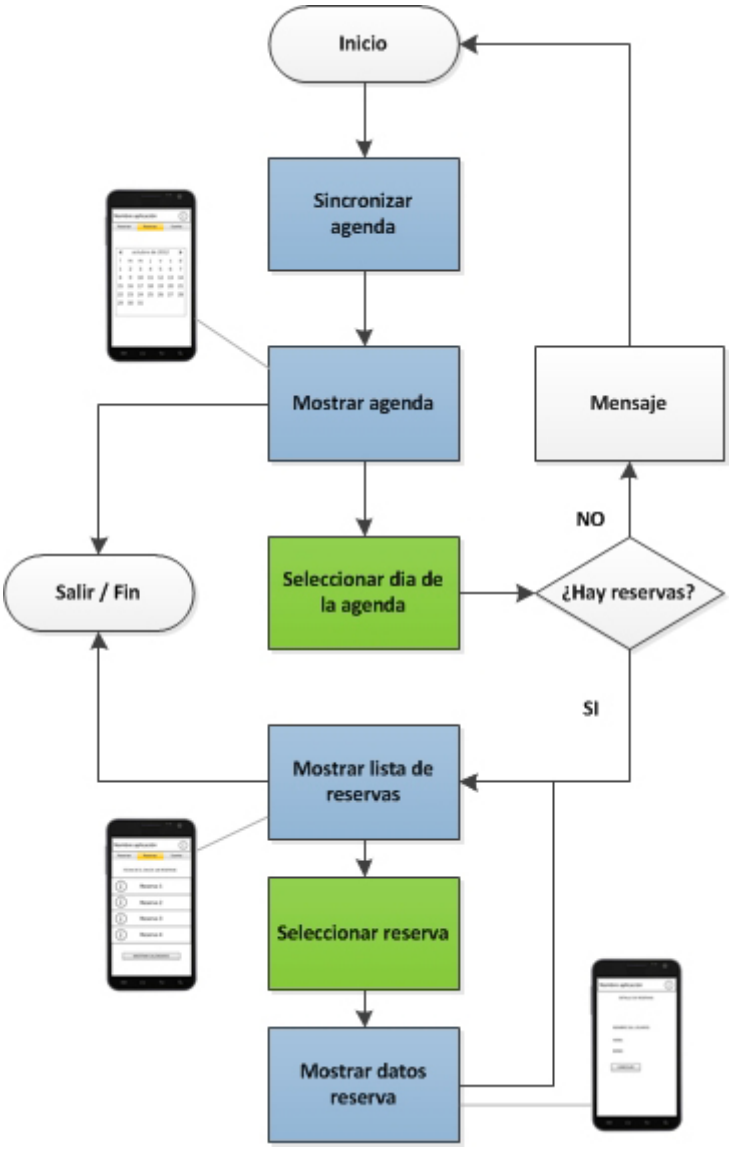


Ilustración 4. Consultar reservas, empresa

Diagrama de flujo realizar reservas.

Este diagrama nos muestra la estructura en el flujo de desarrollo cuando un usuario de tipo empresa quiere realizar manualmente una reserva.

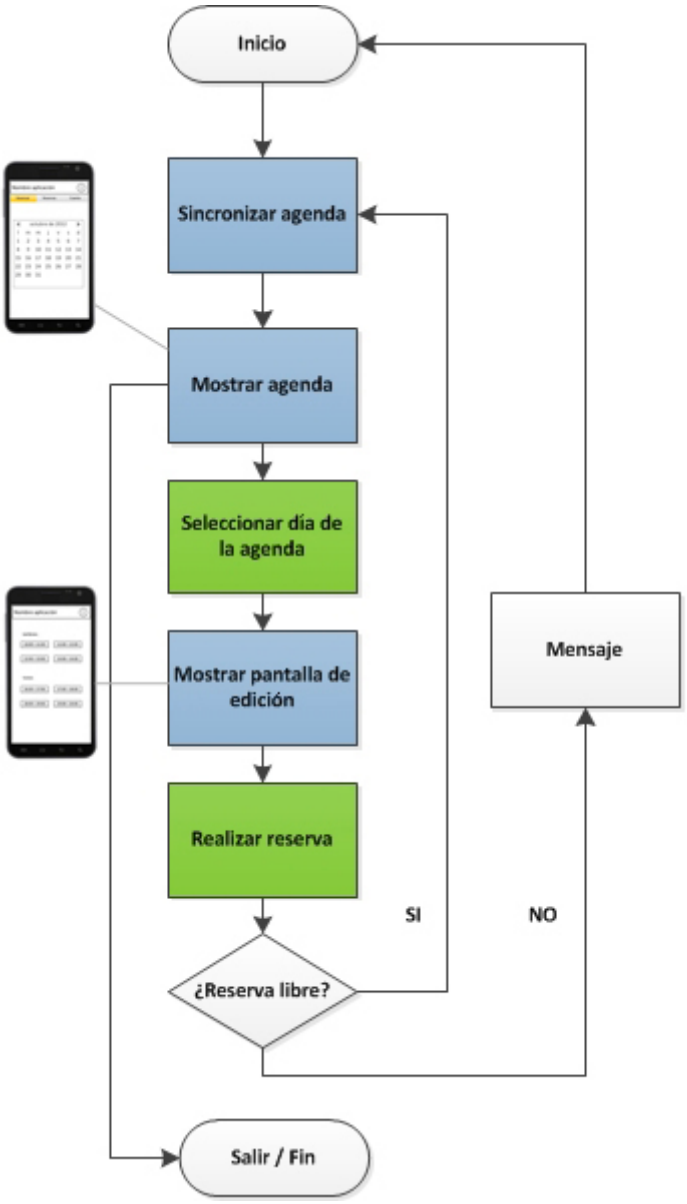


Ilustración 5. Realizar reservas, empresa

Diagrama de flujo editar reservas.

Este diagrama nos muestra la estructura en el flujo de desarrollo cuando un usuario de tipo empresa quiere editar manualmente una reserva.

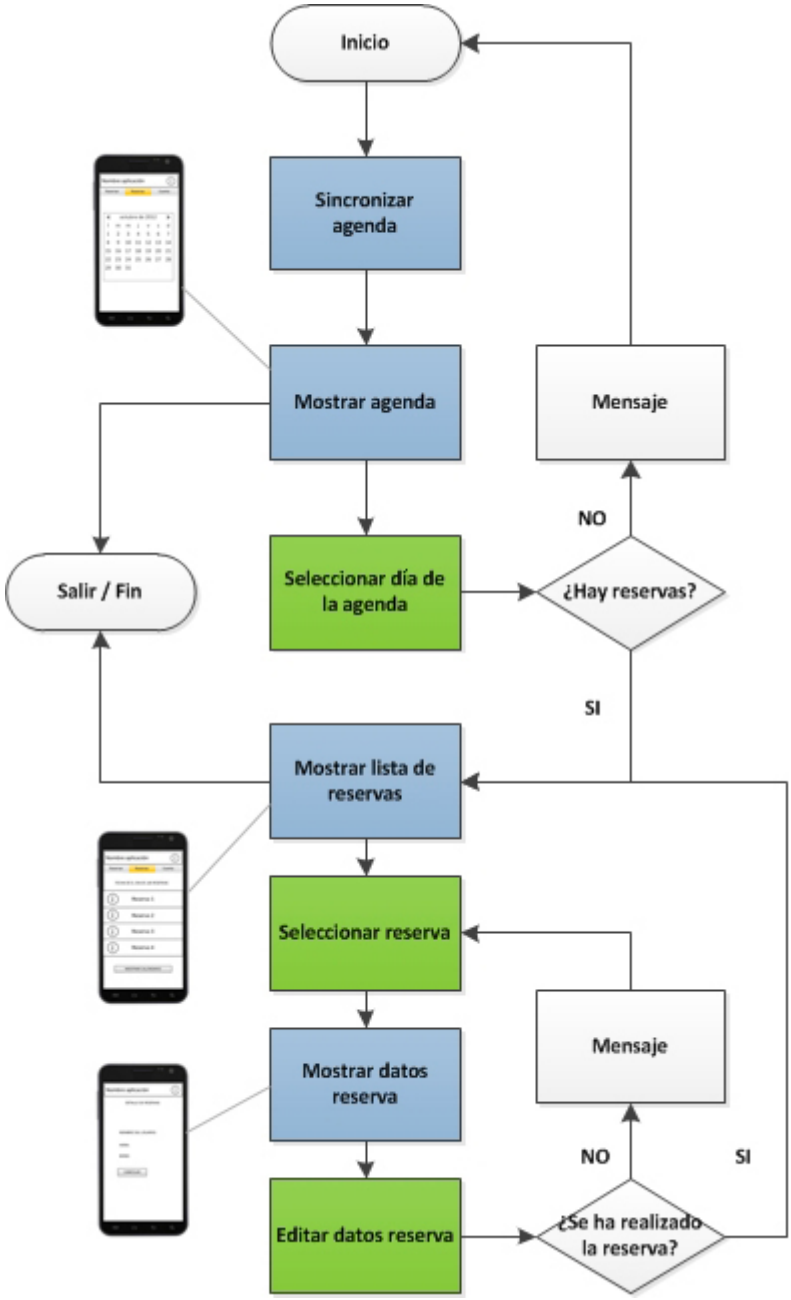


Ilustración 6. Editar reservas, empresa

Diagrama de flujo editar cuenta de usuario.

Este diagrama nos muestra la estructura en el flujo de desarrollo cuando un usuario de tipo empresa quiere editar su cuenta de usuario.

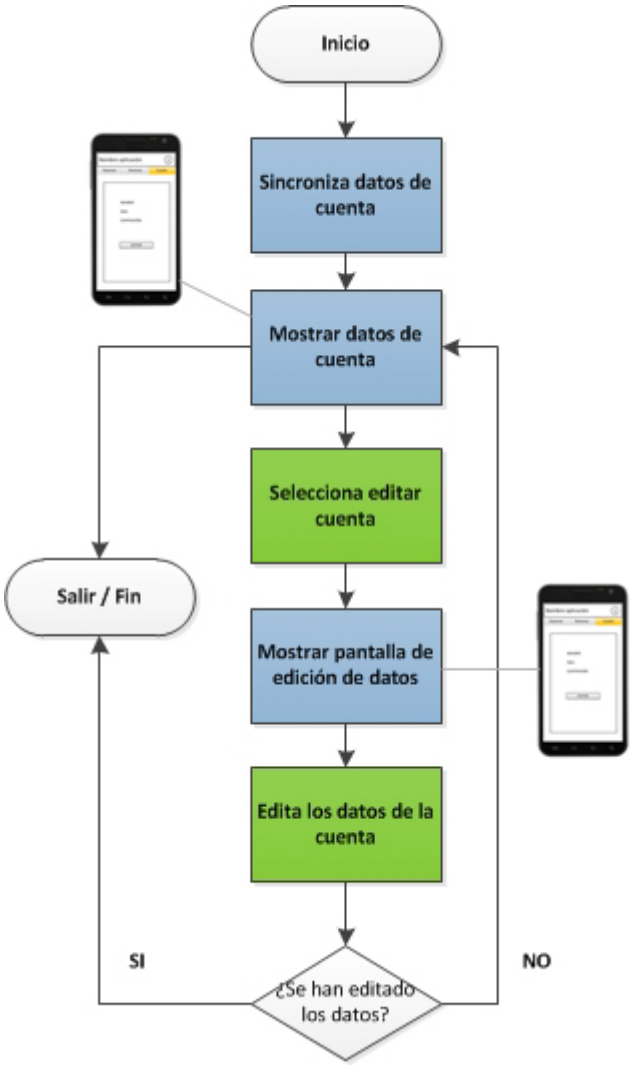


Ilustración 7. Editar cuenta de usuario, empresa

Perfil de usuario cliente

Diagrama de flujo registro de usuario.

Este diagrama nos muestra la estructura en el flujo de desarrollo cuando un usuario de tipo cliente entra por primera vez en la aplicación y se registra como cliente.

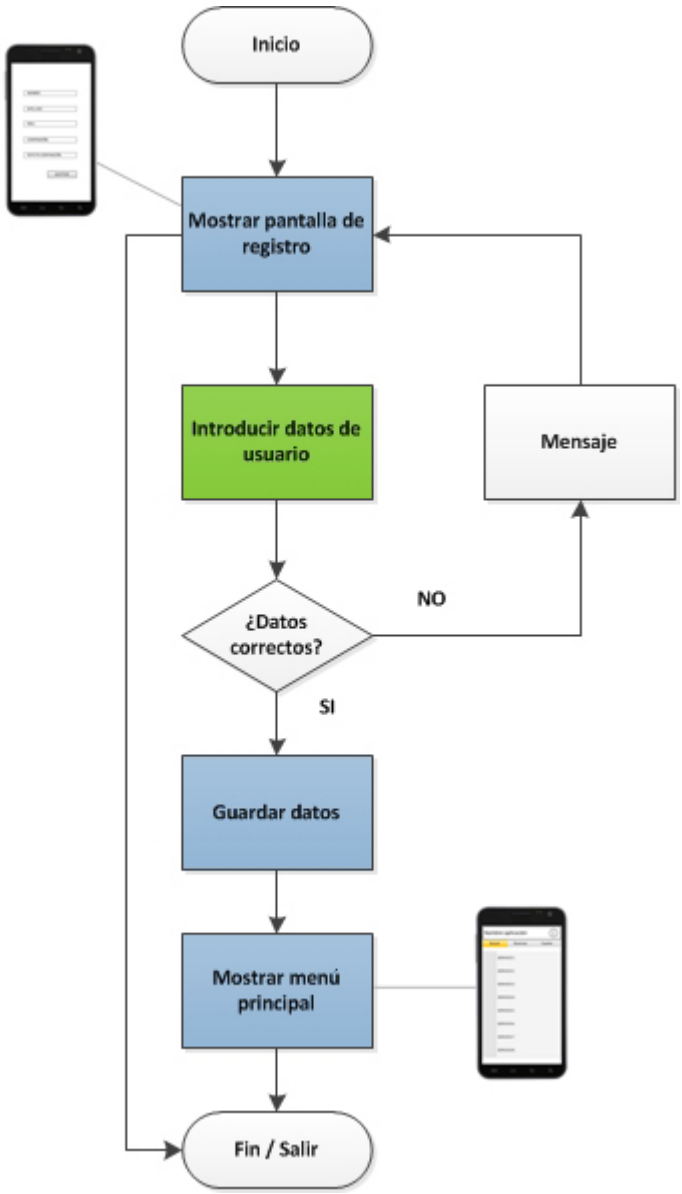


Ilustración 8. Registro de usuario, cliente

Diagrama de flujo consultar reservas.

Este diagrama nos muestra la estructura en el flujo de desarrollo cuando un usuario de tipo cliente quiere consultar las reservas que ha realizado.

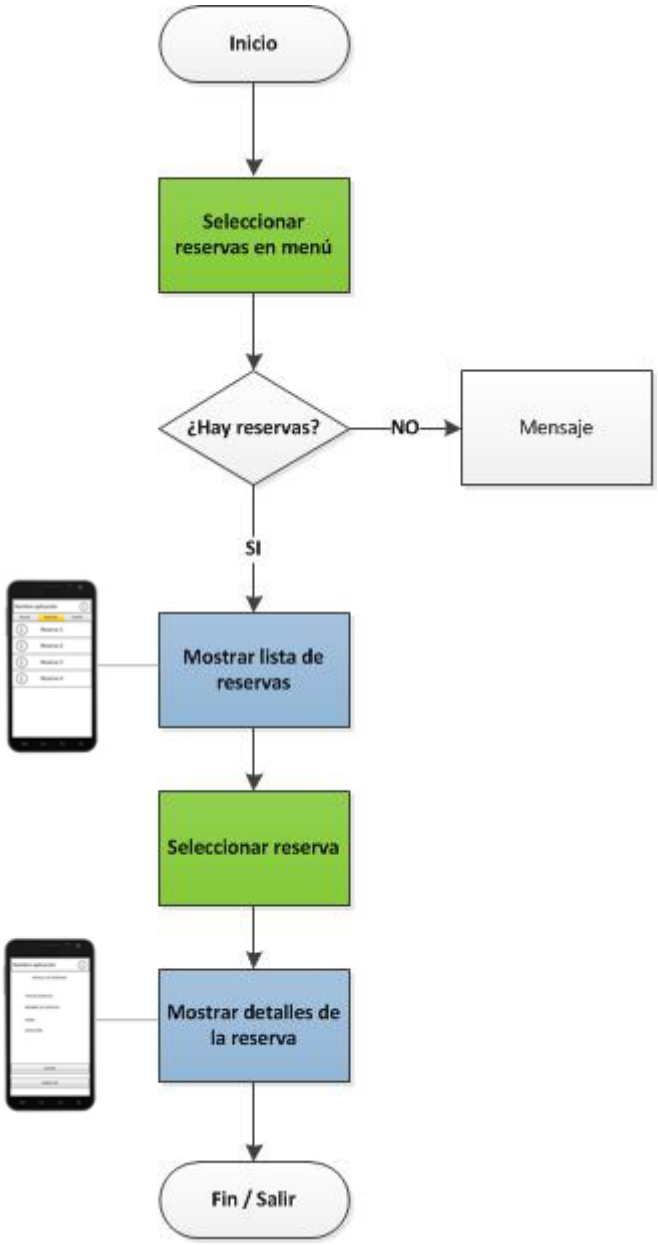


Ilustración 9. Consultar reservas, cliente

Diagrama de flujo realizar reservas.

Este diagrama nos muestra la estructura en el flujo de desarrollo cuando un usuario de tipo cliente quiere realizar una reserva en un servicio.

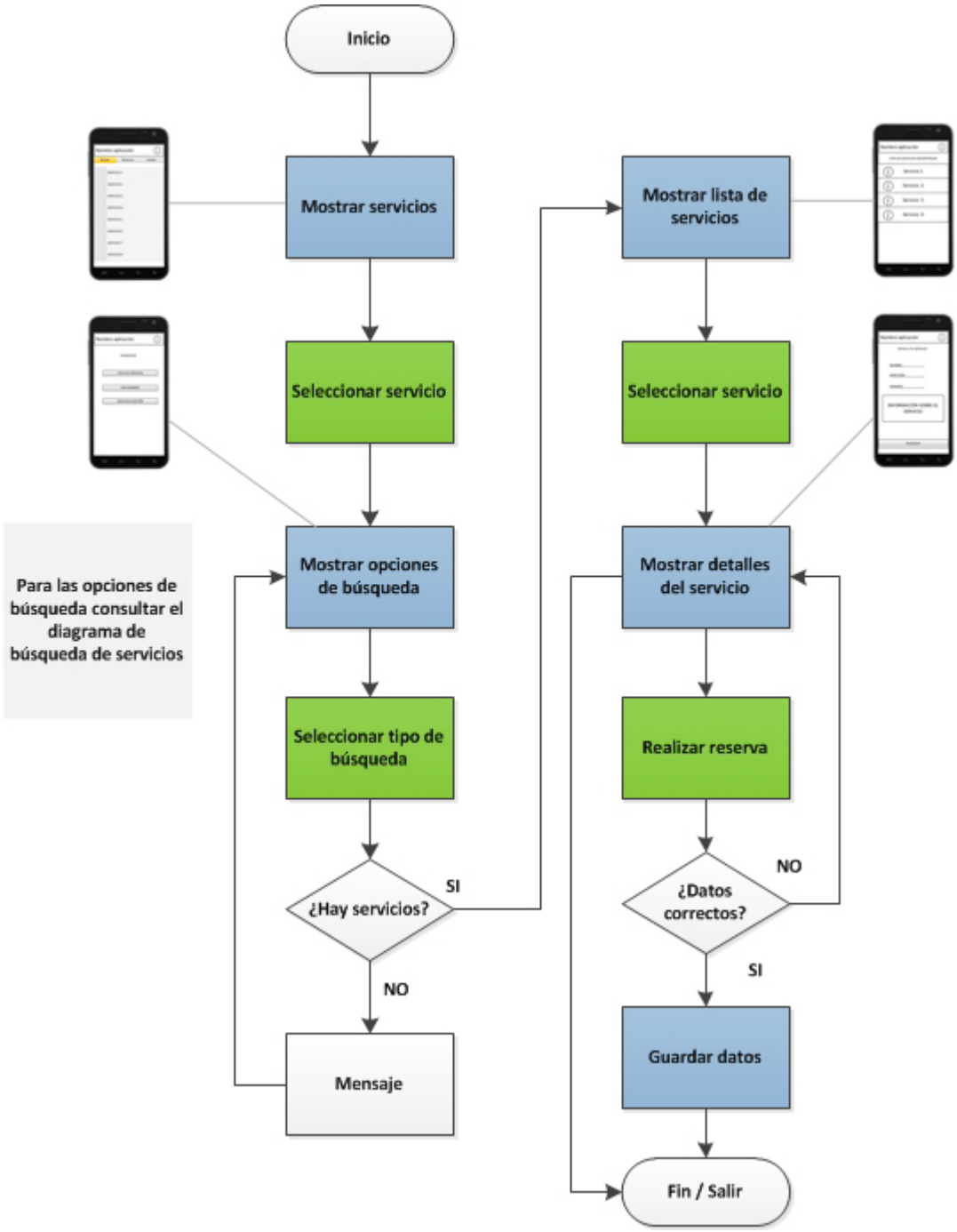


Ilustración 10. Realizar reservas, cliente

Diagrama de flujo editar reservas.

Este diagrama nos muestra la estructura en el flujo de desarrollo cuando un usuario de tipo cliente quiere editar una reserva anteriormente realizada.

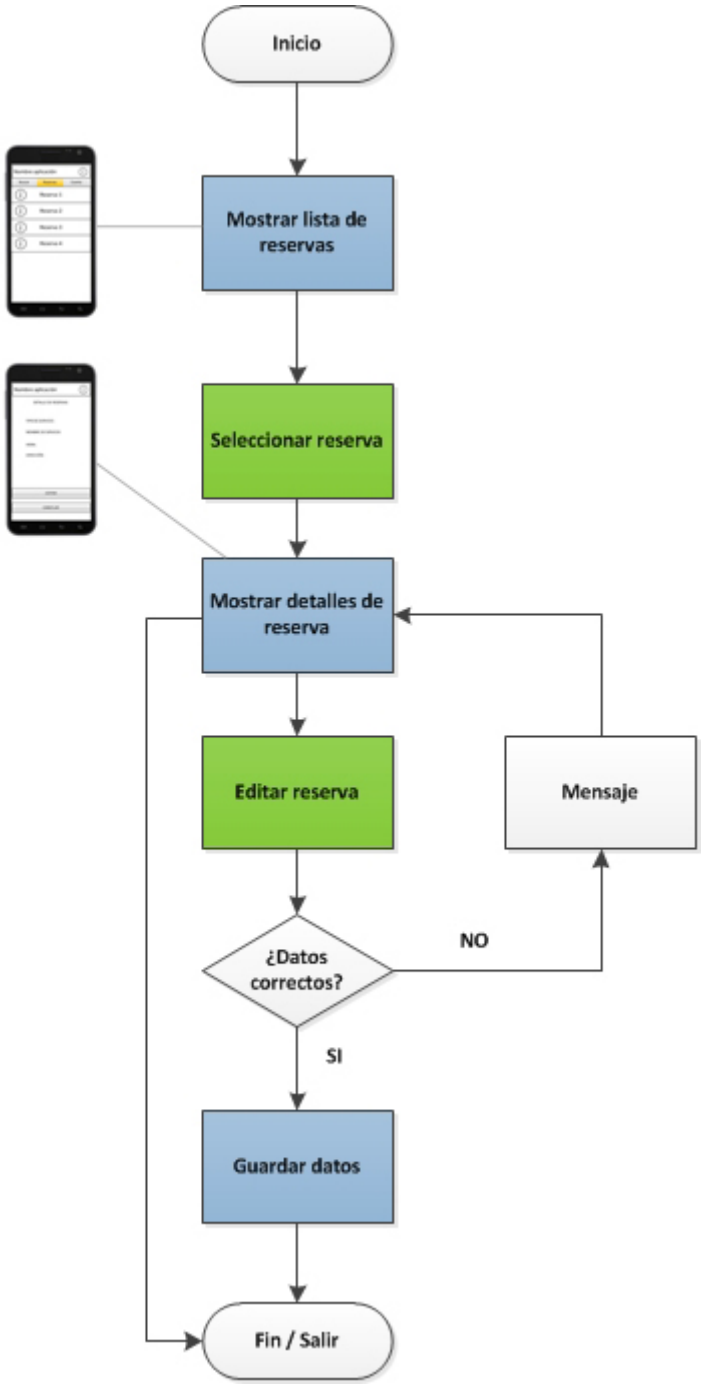


Ilustración 11. Editar reservas, cliente

Diagrama de flujo realizar búsquedas.

Este diagrama nos muestra la estructura en el flujo de desarrollo cuando un usuario de tipo cliente quiere realizar búsquedas de servicios.

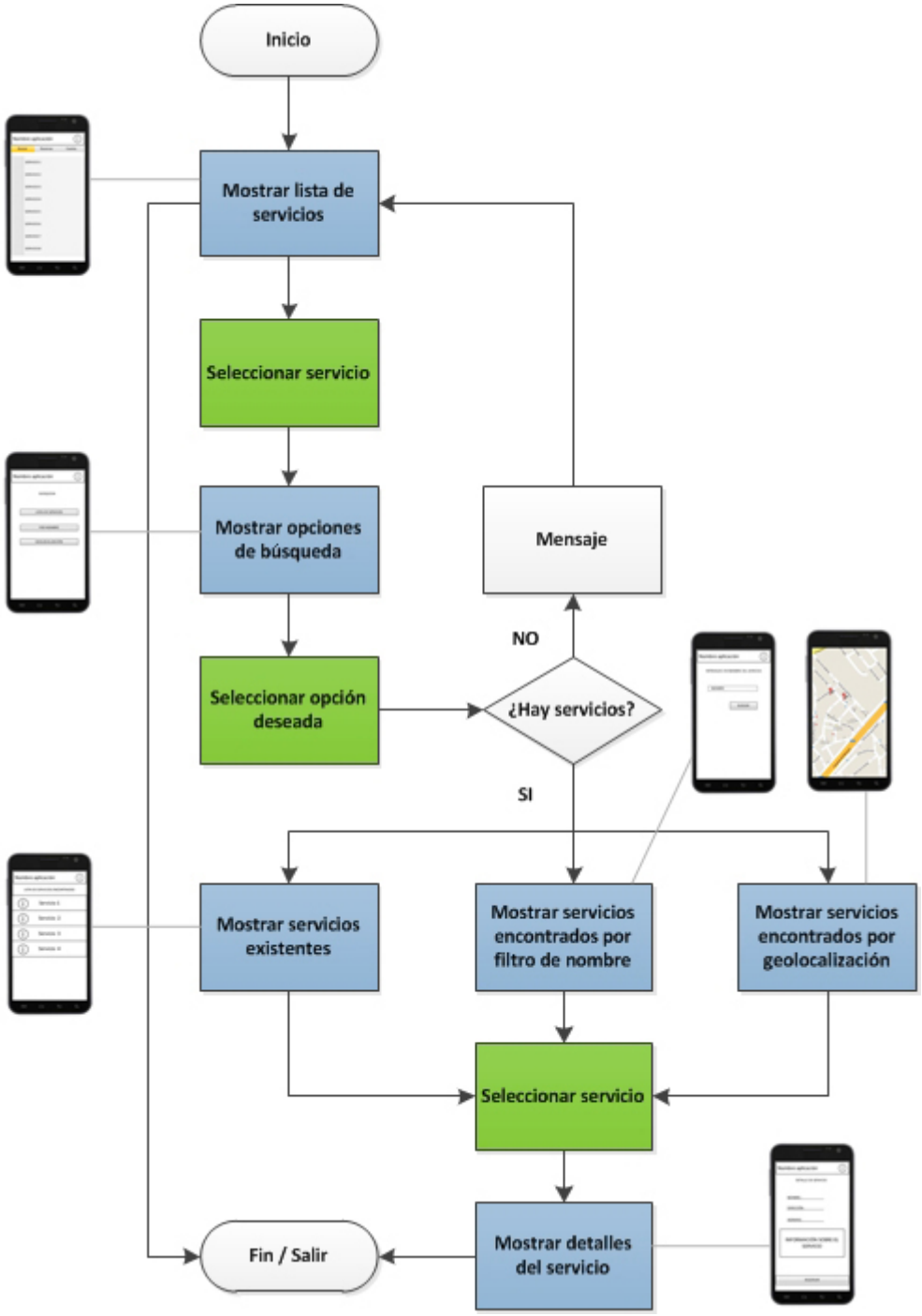


Ilustración 12. Realizar búsquedas, cliente

Diagrama de flujo realizar un recordatorio.

Este diagrama nos muestra la estructura en el flujo de desarrollo cuando un usuario de tipo cliente quiere realizar un recordatorio de reserva sobre un servicio.

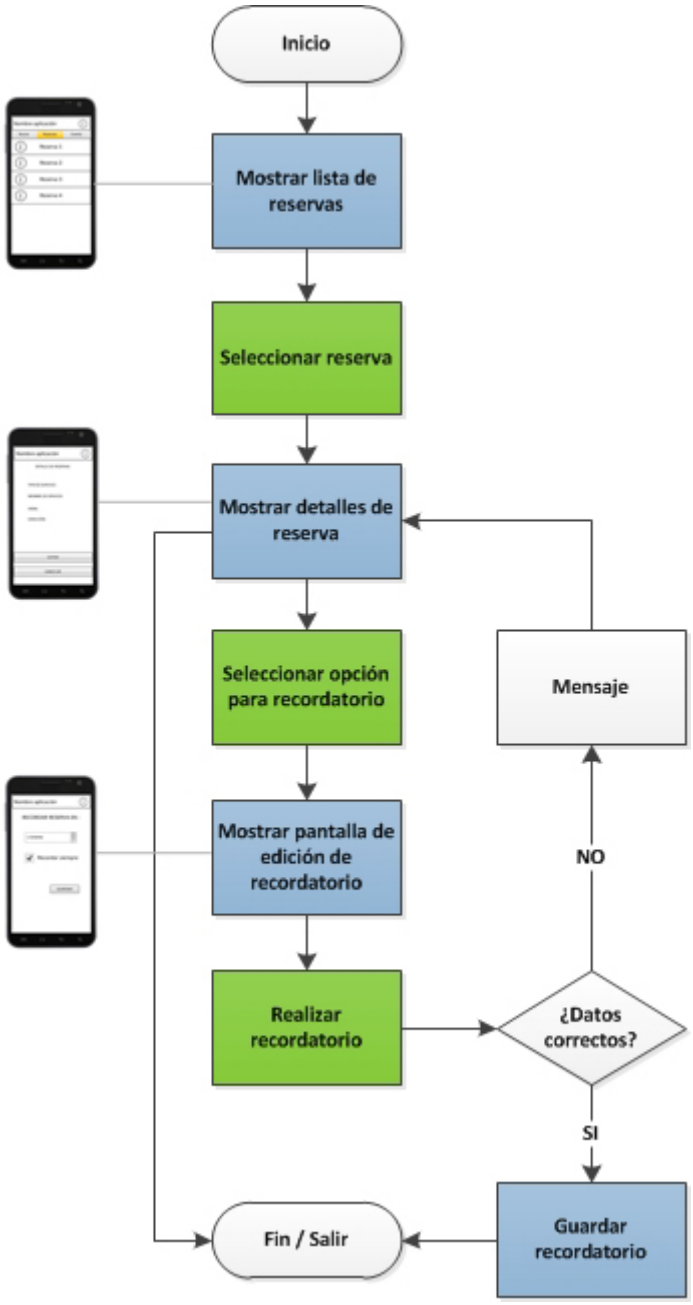


Ilustración 13. Realizar un recordatorio, cliente

Diagrama de flujo editar cuenta de usuario.

Este diagrama nos muestra la estructura en el flujo de desarrollo cuando un usuario de tipo cliente quiere editar su cuenta de usuario.

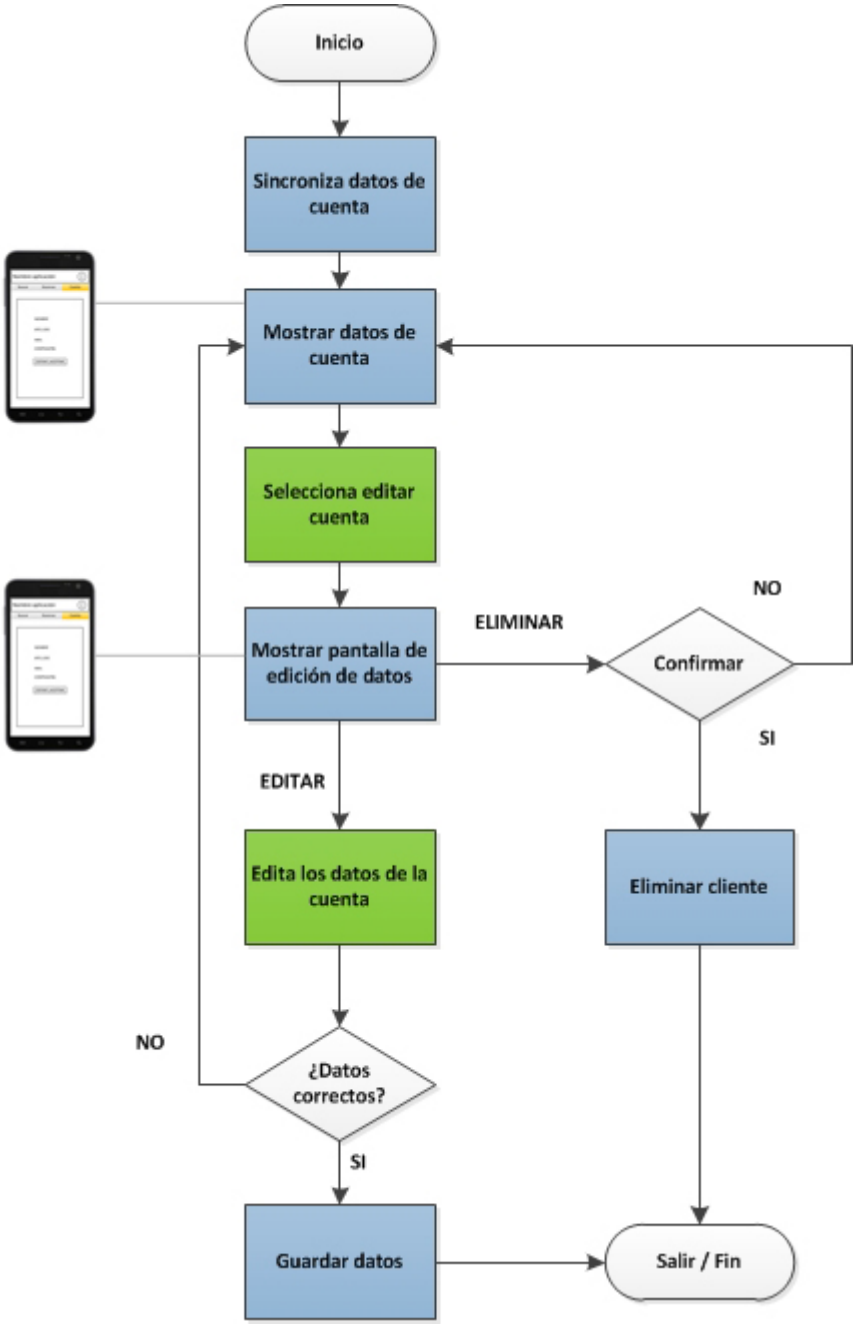


Ilustración 14. Editar cuenta de usuario, cliente

2.5 Prototipo

2.5.1 Primeras ideas de diseño

Estas imágenes muestran las primeras ideas de la estructura y los diseños de las pantallas de la aplicación y como se interactúa a través de ellas.

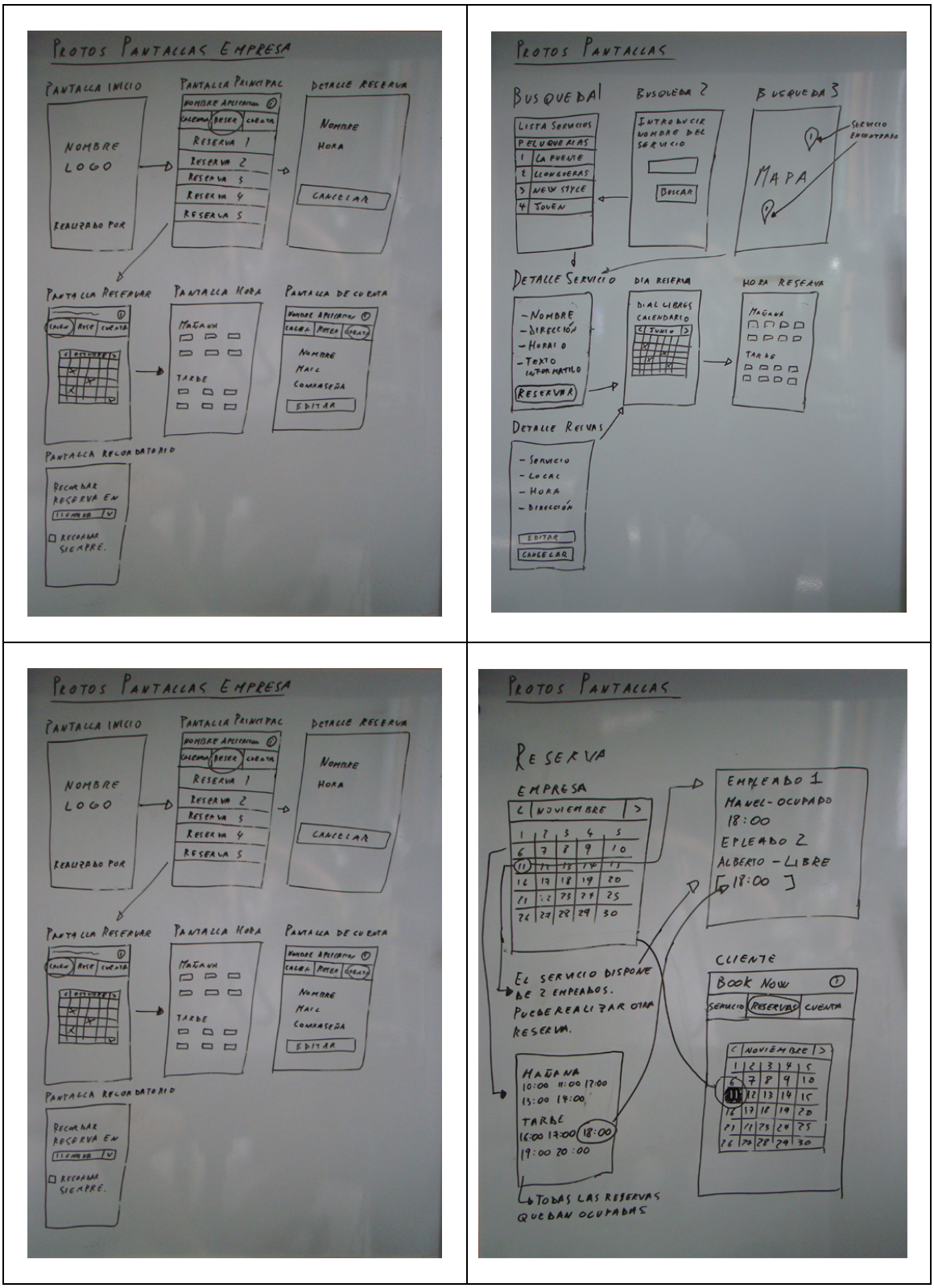


Ilustración 15. Ideas de diseño

2.5.2 Prototipo de pantallas

En este apartado se van a mostrar los prototipos finales para la interface gráfica, tendremos dos partes diferenciadas, unas pantallas para cuando la aplicación la utilice el usuario de perfil empresa y otras para cuando las utilice el cliente.

Esta es una muestra de la presentación de las pantallas que muestra la pantalla común para los 2 perfiles de usuario de inicio de la aplicación:

1 **Pantalla inicial**

Ilustración 16. Pantalla inicial

Pantallas perfil empresa

En este apartado se mostrarán los prototipos de las pantallas que se utilizarán en la aplicación cuando accedamos como empresa.

Organigrama de navegación:

Ilustración 17. Organigrama de pantallas de empresa

Pantallas:

2 Pantalla de inicio de sesión

3 Pantalla de reservas

Ilustración 18. Pantallas inicio de sesión y reservas, empresa

4 Pantalla de cuenta de usuario

5 Pantalla de realización de reservas

Ilustración 19. Pantallas de cuenta de usuario y realización de reservas, empresa

6 **Pantalla de detalles de reservas**

7 **Pantalla de calendario de reservas**

Ilustración 20. Pantallas de detalles de reservas y calendario de reservas, empresa

8 **Pantalla de hora de reserva**

Ilustración 21. Pantalla de hora de reserva, empresa

Pantallas perfil cliente

En este apartado se mostrarán los prototipos de las pantallas que se utilizarán en la aplicación cuando accedamos como cliente.

Organigrama de navegación:

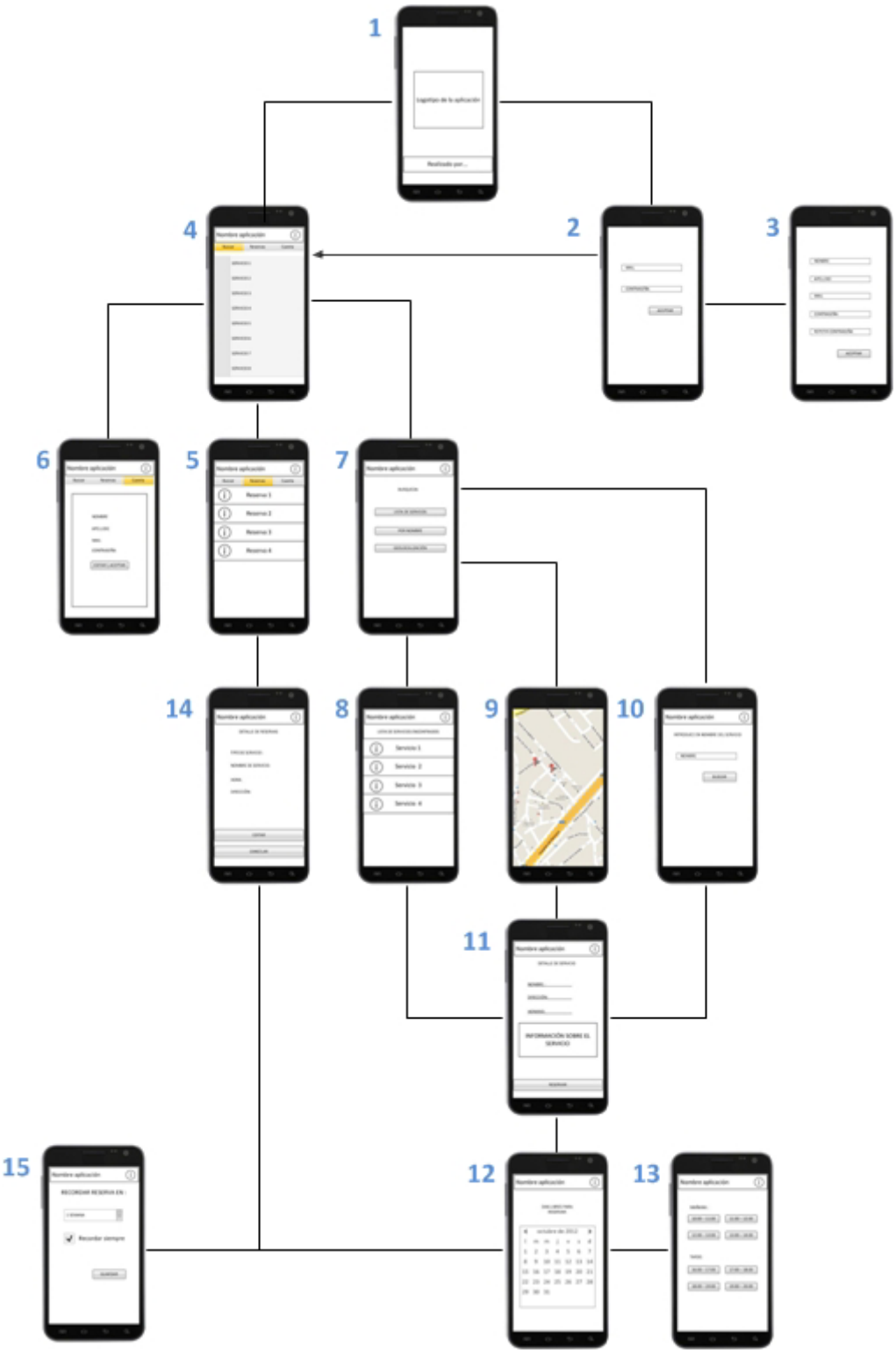


Ilustración 22. Organigrama de pantallas de cliente

Pantallas:

2 Pantalla de inicio de sesión

3 Pantalla de registro de usuario

Ilustración 23. Pantallas de inicio de sesión y registro de usuario, cliente

4 Pantalla de servicios o búsqueda

5 Pantalla de reservas de usuario

Ilustración 24. Pantallas de servicios y reserva de usuario, cliente

6 **Pantalla de cuenta de usuario**

7 **Pantalla de tipo de búsqueda**

Ilustración 25. Pantallas de cuenta de usuario y búsqueda, cliente

8 **Pantalla de servicios encontrados**

9 **Pantalla de localización de servicios**

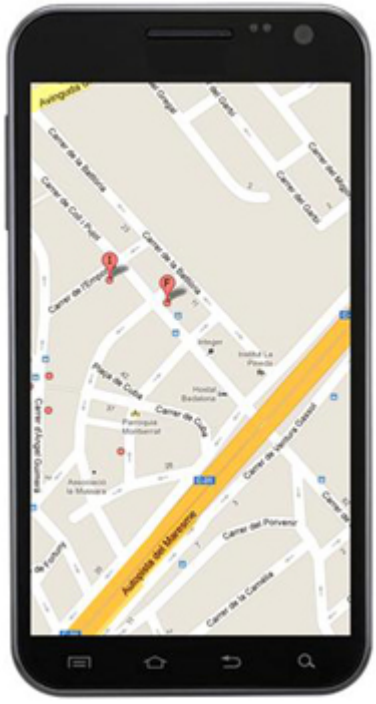


Ilustración 26. Pantallas de servicios encontrados y localización, cliente

10 **Pantalla de búsqueda por nombre**

11 **Pantalla de detalle de servicio**

Ilustración 27. Pantallas de búsqueda y detalle de servicio, cliente

12 **Pantalla de calendario para reserva**

13 **Pantalla hora para reserva**

Ilustración 28. Pantallas de calendario y hora de reserva, cliente

14 **Pantalla de detalle de reserva**

15 **Pantalla de recordatorio**

Ilustración 29. Pantallas de detalle de reserva y recordatorio, cliente

2.5.3 Influencias sobre el diseño.

En este apartado vamos a explicar cómo han afectado las entrevistas y test funcionales a los usuarios para después realizar el diseño principal.

Diseño general.

La idea inicial era una pantalla con un menú de botones para navegar por las diferentes opciones abriendo varias ventanas pero una de las conclusiones que se sacan es que para este tipo de aplicación los usuarios prefieren pantallas sencillas con pocos componentes que no se muestren muy cargadas, intuitivas y que cuando se realicen acciones sea con pocos pasos sin multitud de ventanas.

Para poder cumplir con estos objetivos se decide crear una pantalla principal donde a simple vista se pueda navegar por las tres opciones principales tanto en el modo de cliente o de empresa creando una barra de navegación con tres tabuladores de manera que evitamos tener un menú principal que nos fuerce a navegar por otras pantallas manteniéndonos siempre en la misma y con iconos que resulten intuitivos, además se consigue no cargar la pantalla ya que en cada acción de navegación se mostrará solo la información que queremos ver.

Tenemos una serie de objetos como servicios y reservas que se tienen que poder mostrar de manera que sea fácil navegar para consultarlos y además interactuar sobre ellos, para facilitar esta navegación se decide crear en las pantallas listas verticales y deslizables que muestran iconos o colores representativos y que te permiten seleccionar el objeto correspondiente.

Calendario.

Para seguir con la idea de un diseño fácil y que las acciones se puedan realizar en pocos pasos se pensó en crear un calendario donde a simple vista se puedan ver las reservas realizadas y que sea interactivo para el usuario es decir que desde él se pueda realizar con facilidad una reserva pulsando sobre el día correspondiente, esta era una idea que ya se pensó en el planteamiento inicial y que después de realizar los test los resultados fueron positivos.

Mapa.

Cuando se pensó en el primer diseño de la aplicación ya se planteó la opción de incluir un mapa para que la búsqueda de servicios fuera más sencilla de realizar pero eso suponía invertir más tiempo en una tecnología que tal vez después no se utilizará por el usuario final.

Los test dan un resultado muy favorable a la utilización de mapas para la búsqueda de servicios así que esta opción pasa a ser prioritaria y se incluyen tanto las pantallas de mapa como los controles para acceder al mapa en las pantallas correspondientes.

Horarios.

Para el diseño de los horarios influía un poco la funcionalidad final, una de las opciones que se quieren mejorar de la aplicación es que los horarios no sean fijos si no que se puedan editar, al ser valores dinámicos esto influiría en la manera de representar la información en la pantalla y de interactuar con ella, en los test funcionales la idea de que los horarios sean fijos no resulta

ser un problema para los usuarios, de ese modo en este primer diseño para la pantalla de horarios se diseñan fijos dejando la opción como posible mejora para siguientes versiones de la aplicación.

Detalles servicios.

El cliente de tipo empresa en las entrevistas le da mucha importancia al hecho de poder informar al usuario de promociones, servicios, oferta etc.

De ese modo en la pantalla de destalles de servicios se decide incluir un campo de varias líneas de texto donde pueda salir la información que se quiere mostrar al cliente.

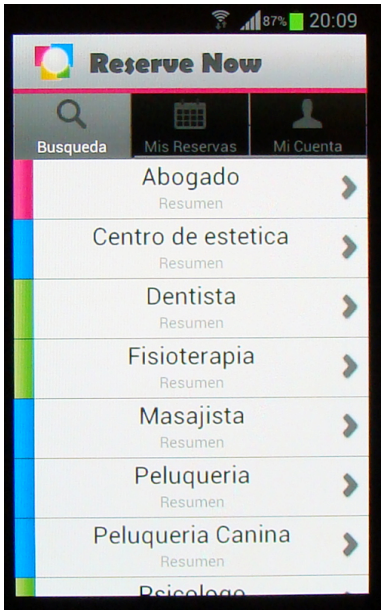
2.5.4 Ideas en la solución de diseño.

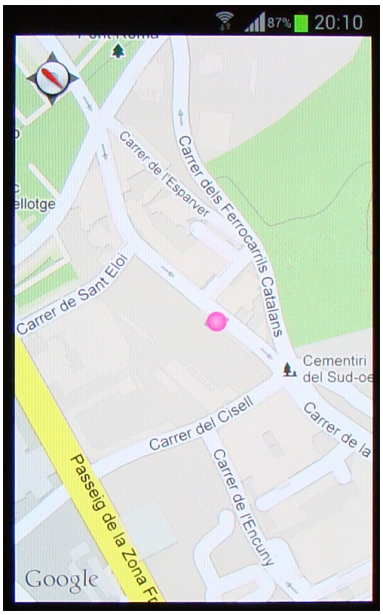
Estas son algunas imágenes de la beta de diseño final de algunas pantallas de la aplicación sobre un dispositivo.

En la pantalla que muestra el menú principal veremos en el diseño que cada servicio está relacionado con un color.

<table>
<tr><td>La idea es que cada color corresponda a un servicio para que el usuario relacione los servicios por colores.

<table>
<tr><td></td><td>Servicios jurídicos</td></tr>
<tr><td></td><td>Estética y spa</td></tr>
<tr><td></td><td>Salud</td></tr>
<tr><td></td><td>Servicios de reparación</td></tr>
</table>
</td><td>Pantalla de menú principal

 
</td></tr>
<tr><td>Pantalla de mapa</td><td>Pantalla de inicio</td></tr>
<tr><td>En el mapa el servicio saldrá reflejado con su color, en ésta imagen mostramos un servicio jurídico.</td><td>En el logotipo se representan los cuatro colores utilizados para representar los servicios.</td></tr>
</table>



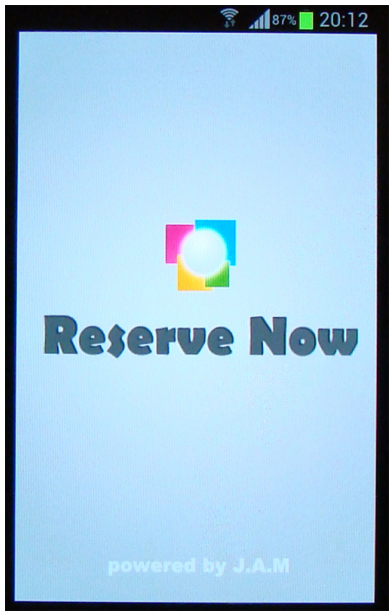


Ilustración 30. Ideas de solución de diseño

3. Arquitectura e implementación

3.1 Introducción

En este apartado se hace una pequeña introducción a la plataforma Android y veremos la arquitectura de la aplicación especificando los diferentes módulos que la forman.

Objetivo:

El objetivo de este apartado es poder tener la arquitectura del proyecto especificada para poder implementar el código de la aplicación y tener un punto de referencia para poder retomar en un futuro el proyecto con facilidad.

3.2 Plataforma Android

La aplicación esta realizada para trabajar sobre Android.

Android es un sistema operativo y una plataforma software en la que se desarrollan aplicaciones que está formado por un robusto entorno de desarrollo de código abierto.

Android fue desarrollado con la idea de ser más tolerable a fallos y creado con una capa de abstracción sobre Linux para desentender al desarrollador de tareas de más bajo nivel y darle más seguridad al sistema.

Para poder ejecutar el código java Android dispone de su propia maquina virtual Dalvik que está expresamente optimizada para dispositivos móviles.

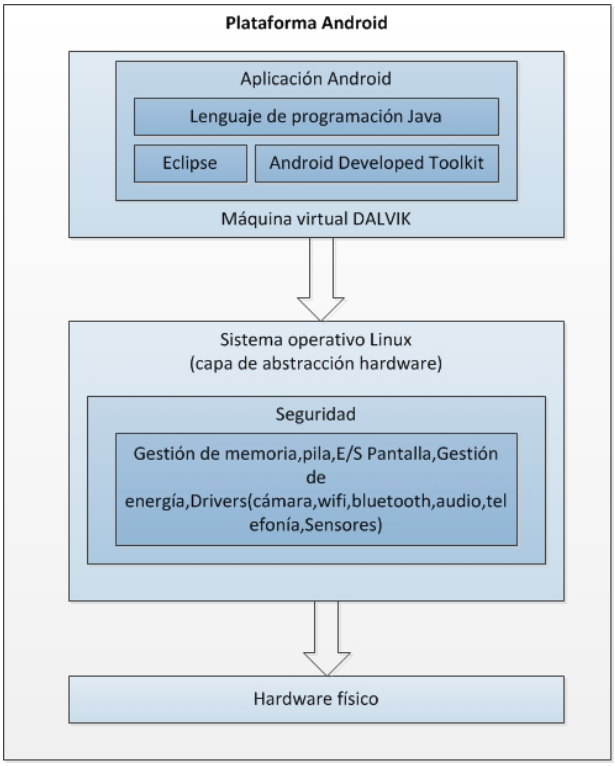


Ilustración 31. Plataforma Android

3.2.1 Entorno de la aplicación Android

El entorno de la aplicación Android proporciona todo lo necesario para implementar una aplicación, el ciclo de vida de una aplicación incluye los siguientes componentes clave:

- Activity o actividades son funciones ejecutadas por la aplicación representan una actividad de la aplicación que puede estar activa o en reposo.

- Los grupos de vistas que definen el diseño de la aplicación.

- Los Intens son mecanismos para cambiar de actividad o para informar al sistema sobre planes de una aplicación.

- Los servicios permiten el procesamiento en segundo plano sin la intervención del usuario.

- Las notificaciones alertan al usuario cuando ocurre algo importante.

- Los proveedores de contenido facilitan la transmisión de datos entre diferentes aplicaciones.

3.2.2 Componentes clave

Dentro de la arquitectura Android hay muchos componentes importantes en este apartado explicaremos los que se han utilizado con más frecuencia en la aplicación desarrollada.

Activity

En una aplicación Android en núcleo principal es la clase activity (android.app.Activity), normalmente definiremos una clase activity para cada pantalla de la aplicación, pero también podemos tener una activity que no requieran de pantalla gráfica.

Este tipo de clases representan la actividad actual en la que nos encontramos y como tal tiene un ciclo de vida, las aplicaciones Android pueden tener varios procesos y el sistema Android permite que tengamos varias aplicaciones ejecutadas concurrentemente siempre que los requisitos de hardware del sistema lo permitan.

Las aplicaciones pueden ejecutarse en segundo plano y pueden ser interrumpidas o detenidas, en el dispositivo solo podemos tener una Activity visible para el usuario a la vez, el resto de objetos Activity abiertas se mantendrán en segundo plano y no serán visibles y siempre dependiendo de los recursos del sistema.

El sistema controla todos los objetos Activity que se están ejecutando dentro de una pila, cuando se inicia una nueva actividad esta se sitúa por delante de la última actividad abierta dentro de la pila y esta se detiene, de esta manera cuando esta última actividad finaliza al anterior que se había detenido se reanuda.

Cada Activity es responsable internamente de gestionarse para llevar esa gestión dispone de unos métodos que son llamados según los estados internos en los que se encuentra la actividad que influyen en ciclo de vida de la misma.

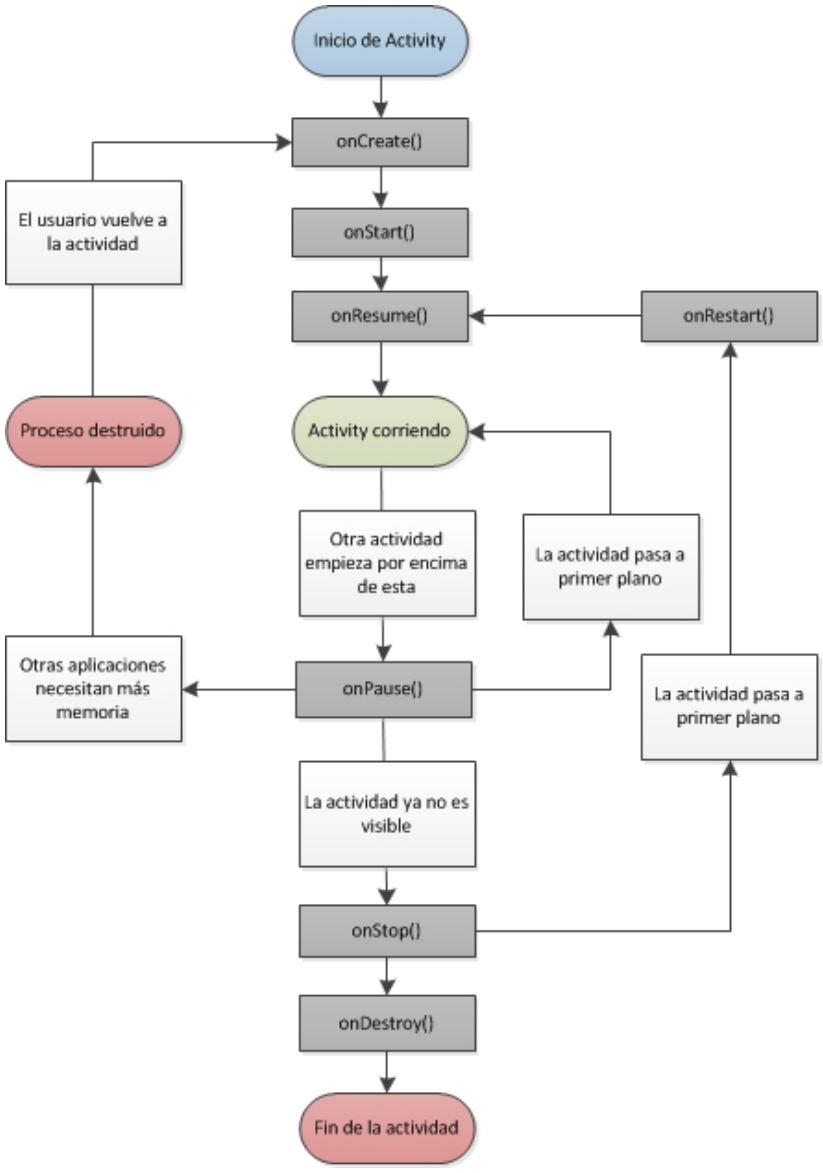


Ilustración 32. Ciclo de vida de una Activity

Intents

Durante el transcurso de una aplicación Android esta puede cambiar entre diferentes instancias Activity, a veces pueden existir varias instancias de una actividad en la pila.

Los Intents nos permiten lanzar nuevas instancias a través del método startActivity() indicándole en nombre la clase Activity que queremos iniciar, podemos usar también el método startActivityForResult() que nos permitirá lanzar una actividad temporal que cuando termine le retorne a la actividad desde donde se ha lanzado un resultado de alguna operación generada en esa actividad.

También nos permiten lanzar mensajes internos en el dispositivo para comunicarnos con otras aplicaciones.

Bundle

El objeto Bundle nos permite guardar la información del estado de una actividad a través del método onSaveInstanceState(), esto es muy útil porque en sistema en ocasiones si tiene por falta de memoria puede cerrar alguna actividad que este en segundo plano, de esta manera podemos volver abrir la actividad recuperando la información de su estado.

También no es muy útil para recuperar información trasmitida a través de las actividades utilizando los Intents.

Grupos de Vistas layout

Como hemos comentado en el apartado de las clases Activity estas suelen ir acompañadas de una pantalla o interface gráfica, Android nos permite diseñar estas pantallas con archivos XML y una serie de etiquetas que nos permiten de manera sencilla crear el diseño de la pantalla. El sistema Android en las librerías del kit de desarrollo incluye un grupo bastante amplio de los controles típicos que necesitaremos para montar nuestra interface, además el kit de desarrollo para Eclipse incluye el visor de edición gráfica Graphical Layout que nos permite ver y editar la pantalla desde un panel grafico.

Servicios

Las aplicaciones pueden interactuar con el sistema operativo y con el hardware interno a través de una serie de gestores. Cada gestor es responsable de mantener el estado de algún servicio interno del sistema. Por ejemplo:

- LocationManager facilita la interacción con los servicios basados en localización disponibles en el dispositivo.

- ViewManager y WindowManager gestionan el funcionamiento básico de la pantalla y de la interface de usuario.

- AccesibilityManager gestiona los eventos de accesibilidad, facilitando soporte para usuarios con discapacidades físicas.

- NotificationManager gestiona los eventos de servicios que se ejecutan en segundo plano para avisar al usuario aunque la aplicación no esté visible.

- AudioManager proporciona acceso a los controles de audio y timbre del dispositivo.

Handler

Este controlador nos permite enviar mensajes a diferentes puntos de nuestra aplicación, es un subproceso que se ejecuta en segundo plano permitiéndonos enviar y recibir mensajes a través de unas funciones internas especificas para este propósito, esto es muy útil para poder comunicar la actividad principal con otras actividades que están en reposo.

Manifest

El archivo manifest es un archivo formateado como XML que debe acompañar a cada aplicación Android. Contiene información importante sobre la misma. Es donde se define el nombre de la aplicación y se da información sobre la versión, así como los componentes por los que está formada, permisos que necesita para ejecutarse y más información sobre su configuración.

3.3 Requisitos de desarrollo

Para desarrollar la aplicación se han utilizado los siguientes recursos:

Herramientas de desarrollo:

Eclipse 3.7.2 con ADT (Android Development Toolkit).

Adobe Photoshop CS5.

Kits de desarrollo:

Android SDK Tools

Android SDK Platforms-Tools

Google APIS 4.1 Level 16

Parse 1.1.4 SDK

Dispositivos donde se ha probado la aplicación:

Samsung Galaxy S2

Sony Xperia U

Samsung Galaxy ACE

3.4 Contenido

La aplicación está formada por los siguientes módulos distribuidos en los siguientes directorios con el código fuente de la aplicación y los archivos que la complementan.

Directorio bin: Binarios para la instalación de la aplicación.

Directorio gen: Archivos de generación principal R.java

Directorio libs: librerías utilizadas

Directorio res: Recursos de aplicación, layouts y values.

Directorio src: Código fuente de la aplicación.

3.5 Librerías principales

Google Api

Son un conjunto de librerías que facilitan el desarrollo con google maps y otras Apis de Google y servicios. Hemos necesitado esta tecnología para poder utilizar el control MapView para representar mapas y puntos de localización en la aplicación.

Para poder hacer uso de esta Api se requiere que aceptemos los términos de uso del servicio para obtener una clase de utilización.

Paquete de soporte Android

Añade varios componentes disponibles en versiones recientes de SDK en las antiguas. Por ejemplo, utilizando este complemento, las API Loader y Fragment introducidas en la API nivel 11 se pueden utilizar en formato compatible hasta la API nivel 4.

Parse

Librerías para la comunicación y utilización del servicio en la nube donde almacenamos los datos remotamente de la aplicación.

3.6 Arquitectura de aplicación

En la arquitectura de la aplicación tenemos una primera capa de interface gráfica que es la capa con la que interactúa el usuario en esta capa se encontrarán todas las pantallas de la aplicación.

La siguiente capa está conectada directamente con la interface gráfica pues ya hemos comentado antes que la mayoría de las actividades están relacionadas con una pantalla, en esta capa es donde se gestionan los eventos producidos en la capa de interface gráfica del usuario y se comunican con la siguiente capa de componentes para realizar las acciones correspondientes, pues en esta capa también tendremos actividades que no dispongan de pantalla gráfica.

En la siguiente capa de componentes tenemos los diferentes módulos con los que trabaja la capa de actividades y a su vez estos requieren las capas inferiores de librerías y persistencia para poder ejecutar los métodos que tiene cada módulo y que se lanzan desde la capa de actividades.

La capa de API y persistencia contienen los métodos necesarios por la capa de módulos para poder utilizar las librerías de más bajo nivel que nos facilitan la utilización de las diferentes tecnologías utilizadas en la aplicación.

Y finalmente tenemos la capa servicios que incluye todos los servicios que nos proporciona el sistema operativo Android para poder utilizar los gestores que interactúan con el sistema operático y con el hardware.

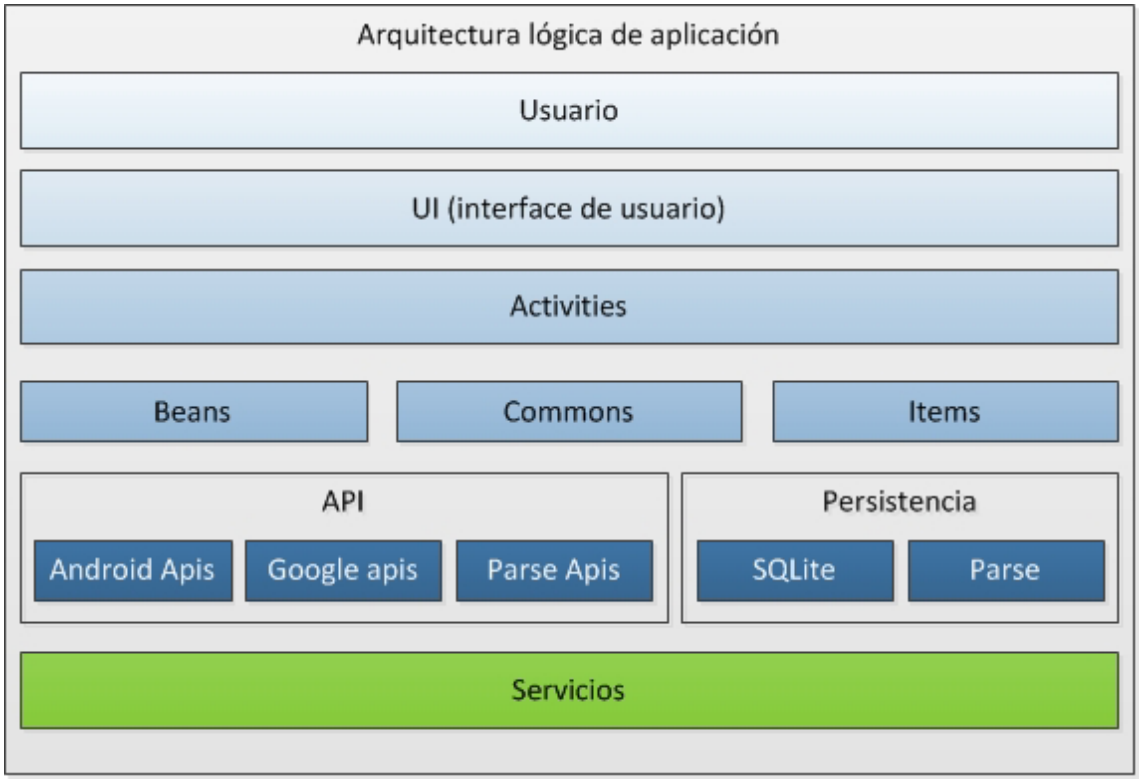


Ilustración 33. Arquitectura aplicación

Un ejemplo en el funcionamiento de la arquitectura sería el siguiente :

El usuario pulsa un botón donde intervendría la capa de interface gráfica, este botón lanzaría su método interno que se encontraría dentro de la capa de actividades y podría ser por ejemplo que mostrara el mapa, para llevar eso a cavo se llamaría al componente correspondiente dentro de la capa de componentes que se encarga de mostrar el mapa con la información correspondiente.

El componente que se encarga de mostrar el mapa hace uso de las Apis de Google para poder utilizar el componente MapView así pasaríamos a la capa de API.

Finalmente las librerías de Google necesitarán usar el Gestor LocationManager para poder tener acceso a los proveedores del dispositivo que nos facilitarán las coordenadas en las que nos encontramos para eso se necesitará llegar hasta la capa de servicios.

3.7 Módulos de aplicación

En este apartado se comentaran los diferentes módulos que forman la aplicación.

La aplicación está repartida en 5 módulos **Activities, Beans, Commons** , **Items** y **Services** , a parte de los directorios comunes de un proyecto Android donde destacaremos la carpeta de contenidos **Layout** y de valores **Values**.

Para consultar la especificación de los métodos internos de cada modulo consultar el Apéndice B.

3.7.1 Activities

Directorio de archivos **src.com.tfc.Ativities**.

En este módulo están todas las clases de actividad que heredan de la clase padre Activity que intervienen en la aplicación y que comentaremos individualmente.

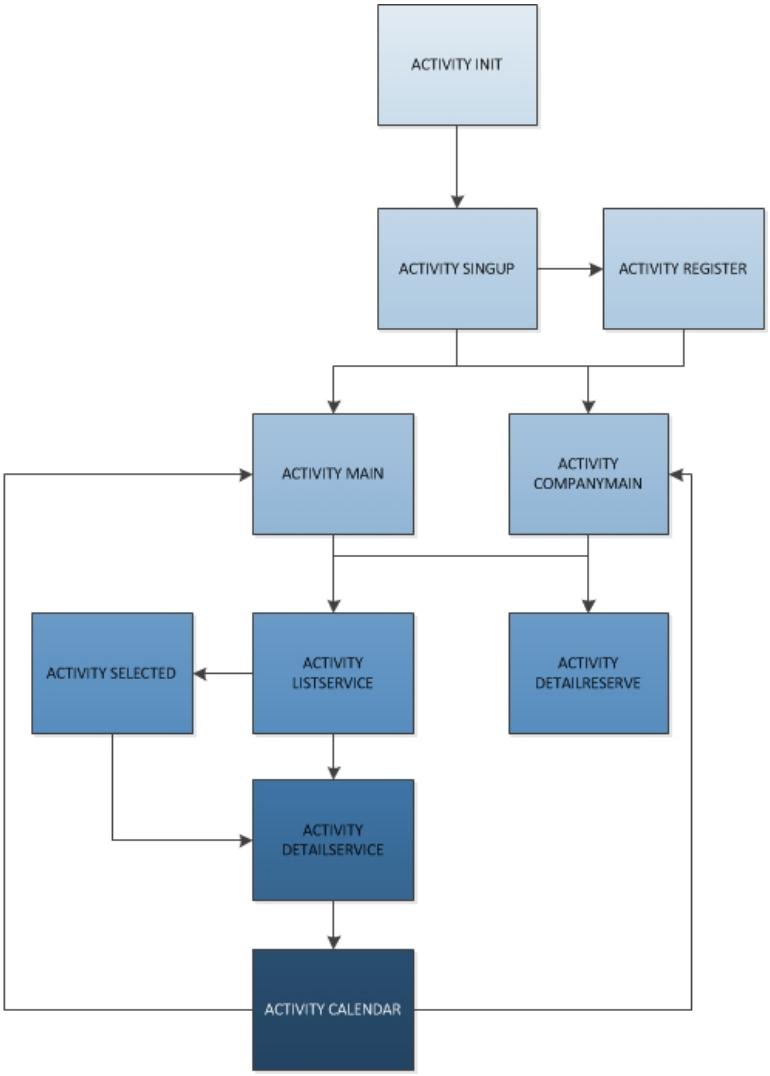


Ilustración 34. Diagrama de actividades

Actividad Inicial

Esta es la actividad donde arranca la aplicación, muestra la pantalla de presentación, inicializa la librería Parse (librería de uso para la persistencia de datos en la nube, Cloud Computing) y se inicializa la sesión de último usuario si la había.

Si hay una sesión iniciada la actividad consulta el usuario de sesión, en función del tipo de usuario si es un cliente o una empresa ejecuta la actividad correspondiente.

Si no hay ninguna sesión abierta carga la actividad de inicio de sesión.

Una vez cargada la actividad correspondiente esta finaliza.

Actividad de Inicio de sesión

Esta actividad es la responsable del inicio de sesión del usuario, carga la pantalla de inicio de sesión y en función del tipo de usuario iniciado inicializa la aplicación en modo cliente o empresa.

Una vez iniciada la sesión esta actividad deja al usuario con la sesión actual abierta y guarda sus datos en la base de datos interna de la aplicación para tener los datos persistentes sobre el último usuario que ha iniciado sesión.

Desde esta actividad puedes acceder a la actividad de registro para dar de alta un nuevo usuario.

Actividad de registro de usuario

Esta actividad nos permite dar de alta un nuevo usuario de tipo cliente, carga la pantalla para la inserción de datos del cliente y registra el nuevo usuario.

Si el registro ha sido correcto inicializa la actividad principal de usuario cliente y esta finaliza.

Actividad principal de usuario cliente

Es la actividad principal de la aplicación cuando entramos en modo de usuario cliente, se conecta con el servidor para recuperar la información de servicios y la información del usuario.

Esta actividad está dividida en tres partes, búsqueda de servicios, reservas de usuario y cuenta de usuario.

Desde esta actividad podemos buscar servicios para los que deseemos tener información o realizar una reserva, consultar nuestras reservas y consultar o editar nuestra cuenta de usuario.

Actividad principal de usuario empresa

Esta es la actividad principal cuando entramos en modo de empresa, se conecta con el servidor para recuperar los datos de usuario y mostrarlos en pantalla.

Esta actividad está dividida en tres partes, agenda de reservas, registro de reservas y cuenta de usuario.

Nos permite consultar las reservas que tenemos para el día que le especificamos a través del calendario y consultar en el calendario como tenemos la agenda, realizar nuevas reservas y consultar o editar la cuenta de usuario.

Utiliza en módulo MyCalendarAdapter para montar el calendario con los datos relacionados del usuario.

Actividad de lista de servicios

Esta actividad nos muestra la lista de servicios relacionada con el tipo de servicio establecido, por ejemplo si seleccionamos el servicio de peluquerías en esta actividad se mostrarán todos los servicios que sean peluquerías.

Podemos filtrar la búsqueda por nombre o lanzar la actividad SelectedActivity que nos mostrará los servicios cercanos representados en un mapa para conocer su localización.

Desde esta actividad seleccionaremos el servicio que queramos consultar.

Actividad de detalla de servicio

Esta actividad nos muestra la información detallada del servicio seleccionado, desde esta pantalla podemos realizar una reserva para este servicio.

Actividad de detallada de reserva

Esta actividad nos muestra la información detallada de la reserva seleccionada, desde esta actividad podemos crear un recordatorio para próximas reservas, eliminar la reserva o visualizar la localización geográfica en un mapa que nos indicará donde está el servicio para el que hemos realizado la reserva.

Actividad de calendario

Esta actividad podemos seleccionar el día y la hora que deseamos para realizar una reserva y completar el proceso de reserva.

El calendario que se representa en pantalla se monta con el módulo MyCalendarAdapter y con los datos de usuario y de servicio para poder indicarnos los días y horas ocupadas y donde ya tenemos reservas.

No nos permitirá reservar en días completos en horas completas o días donde el servicio no abra.

Las horas ocupadas van en función de la cantidad de empleados del servicio.

Actividad de selección geográfica

Esta actividad se encarga de representar en un mapa puntos con ayuda de variables de localización geográfica las posiciones para servicios o usuarios, de esta manera podemos controlar donde están los servicios y a qué distancia se encuentran de nuestra posición.

3.7.2 Beans

Directorio de archivos **src.com.tfc.Beans**.

En este módulo están todos los objetos de datos de uso general del proyecto y realiza funciones de edición sobre el objeto.

Objeto Reminder

Este objeto guarda los datos temporales de los recordatorios en uso por la aplicación.

Objeto Reserve

Este objeto guarda los datos temporales de las reservas en uso por la aplicació

Objeto User

Este objeto guarda los datos de usuario que ha iniciado sesión en la aplicación.

3.7.3 Commons

Directorio de archivos **src.com.tfc.Commons**.

En este módulo están todas las clases de uso común por el resto de clases del proyecto.

Clase común activityMessage

Cada actividad tiene un objeto Handle que se utiliza como mensajería para emitir y recibir los mensajes de esta actividad.

El objeto es común en todas las actividades pero configurado según las necesidades de cada actividad con el tipo de mensajes que se recibe.

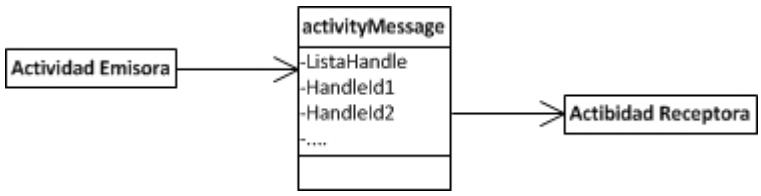
Este es el objeto creado en la actividad DetailServiceActivity que recibe un mensaje que informa que la actividad tiene que finalizar.

```
static  public Handler updateMessage = new Handler() {

      @Override
      public void handleMessage(Message msg) {

            if(msg.arg1 == activityMessage.MSG_ACTIVITY_FINISH)
            {
                  ((DetailServiceActivity) thisActivity).finish();
            }
      }
};
```

La clase activityMessage se encarga de guardar estos objetos en una lista identificándolos con un id para conocer de que actividad procede cada objeto y posteriormente poder usar el Handle desde cualquier actividad indicando solo el identificador de la actividad a la que se dirige y el tipo de mensaje.



El cuerpo del envío tiene el siguiente aspecto:

```
activityMessage.SendMessage(activityMessage.MSG_ACTIVITY_FINISH, 0,
activityMessage.ID_DETAIL_SERVICE);
```

Clase común Constants

Esta clase guarda las variables constantes de uso común en el proyecto, esta clase no tiene ninguna función.

Clase común dialogMessage

Esta clase crea un diálogo informativo para mostrar información al usuario sobre algún evento producido en la aplicación.

Clase común driverDBSQLite

Esta clase es el driver de comunicación con la base de datos SQLite, se utiliza para realizar las consultas sobre la base de datos.

Clase común Session

Esta clase estática se encarga de guardar la referencias de la sesión iniciada por el usuario, guarda una referencia al objeto de la clase usuario y las referencias temporales de los diferentes objetos que se utilicen en la sesión como servicios y reservas para poder ser consultadas desde cualquier módulo del proyecto.

Clase común Tools

Esta clase está compuesta de diferentes funciones útiles para cálculos como comprobar fechas o recuperar valores.

Clase común Geo

Esta clase se encarga de iniciar el servicio de localización geográfica configurarlo y consultar sus parámetros internos.

3.7.4 Items

Directorio de archivos **src.com.tfc.Items**.

En este módulo están todas las clases de tipo ítem, son clases que heredan normalmente de otras clases padre para poder sobrescribir sus métodos y adaptarlas a nuestras necesidades como adaptadores, controles, etc.

Clase item ItemReserveAdapter

Esta clase es un adaptador para editar el aspecto gráfico de un control ListView.

Clase item ItemService

Esta clase es el tipo de objeto que guardará los datos que mostraremos en cada posición en el adaptador de la clase ItemReserveAdapter.

Clase item MyCalendarAdapter

Esta clase es un adaptador para un control GridView que monta un calendario a partir de la fecha actual y los datos cargados en el adaptador que indican que días hay reservas, que días están ocupados y que horarios hay disponibles.

Clase item MyOverlayItem

Esta clase hereda de la clase overlayItem clase que permite dibujar puntos sobre el mapa en superposición, guarda internamente el servicio relacionado con el punto a mostrar en el mapa.

Clase item MyPoint

Clase que hereda de la clase ItemizedOverlay creada para objetos MyOverlayItem, es una lista que guarda los objetos MyOverlayItem relacionándolos con el tipo de datos pasados por parámetros al crear un objeto MyPoint.

3.7.5 Services

Directorio de archivos **src.com.tfc.Services**.

En este módulo esta los servicios de aplicación.

Clase de servicios NotificatiónService

Este servicio se arranca cuando iniciamos el móvil y realiza una comprobación periódicamente para ver si tenemos algún recordatorio cuando tenemos algún recordatorio lanza una notificación desde la que podemos abrir la aplicación.

Clase de servicios BootServiceManager

Esta clase se encarga de cargar el servicio NotificationService para que arranque siempre cuando se inicie el dispositivo después de la carga del sistema operativo.

3.7.6 Layouts

En este apartado comentaremos los archivos layouts que no corresponden a actividades y aun no han sido nombrados en este documento.

Book_name.xml

Interface gráfica del cuadro de diálogo para poner el nombre a las reservas que realiza en usuario de tipo empresa.

Calendar_dialog.xml

Interface gráfica para implementar un calendario en un cuadro de diálogo.

Infobook_detail.xml

Interface gráfica para mostrar la información de una reserva que se acaba de realizar.

List_item_services_layout.xml

Interface gráfica utilizada para la edición de los controles ListView.

List_reminder_dialog.xml

Interface gráfica para mostrar la lista de recordatorios en un diálogo.

Reminder_dialog.xml

Interface gráfica para mostrar los controles que nos permiten realizar un recordatorio en un diálogo.

Time_dialos.xml

Interface gráfica para mostrar las franjas horarias para realizar una reserva en un diálogo.

3.7.7 Values

En este apartado comentaremos los diferentes archivos *.xml que guardan información sobre recursos de la aplicación.

Array.xml

Guarda los datos para cargar la información de control Spinner para el diálogo de creación de recordatorio.

Colors.xml

Guarda la información de los colores que se utilizan en la aplicación.

Strings.xml

Guarda la información de todos los textos que se utilizan en la aplicación.

Styles.xml

Guarda la información del estilo utilizado en la aplicación.

3.8 Persistencia

En este apartado se va a explicar la persistencia de los datos con los que trabaja la aplicación tanto localmente como remotamente.

3.8.1 Persistencia Local

Cuando iniciamos una nueva sesión de usuario este queda registrado en el servidor remoto pero también necesitamos guardar sus datos de forma local para que la siguiente vez que iniciemos la aplicación sepamos quien es el último usuario que inició la sesión y si mantiene la sesión abierta.

Estos datos de usuario los guardamos en una base de datos SQLite que se crea la primera vez que registramos un usuario en la aplicación.

La base de datos solo tendrá una tabla del tipo de objeto User.

Clase User
String userName – Login de usuario. **String** Name – Nombre de usuario. **String** surName – Apellido de usuario. **String** Email – Email de usuario. **String** Type – Tipo de usuario cliente o empresa. **String** Password – Contraseña de usuario. **int** iSession – Marca de sesión abierta. **String** ObjectId – Identificador del objeto

El parámetro iSession nos marcará si la sesión está abierta o cerrada para la próxima vez que iniciemos la sesión.

En la base de datos siempre habrá guardado un solo registro para esta tabla y será el del último usuario que ha utilizado la aplicación.

3.8.2 Persistencia Remota

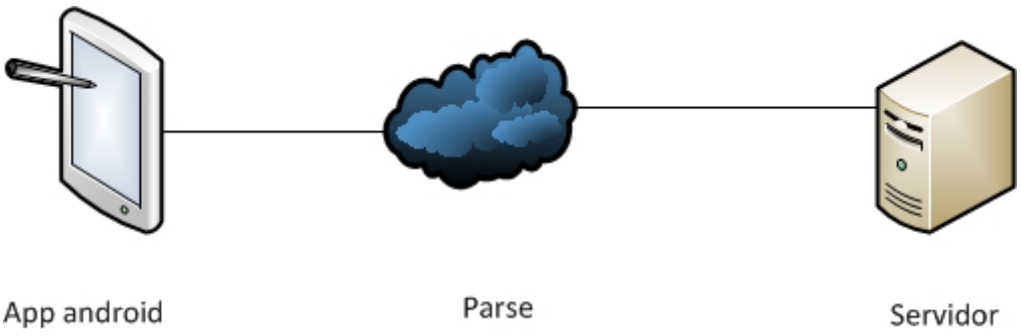


Ilustración 35. Esquema de persistencia remota

Para el sistema de persistencia remota se ha recurrido a un servicio en la nube (Could computing) que nos ofrece una librería de comunicación propia y espacio en servidor gratuito mientras no sobrepasemos el número de consultas mensuales.

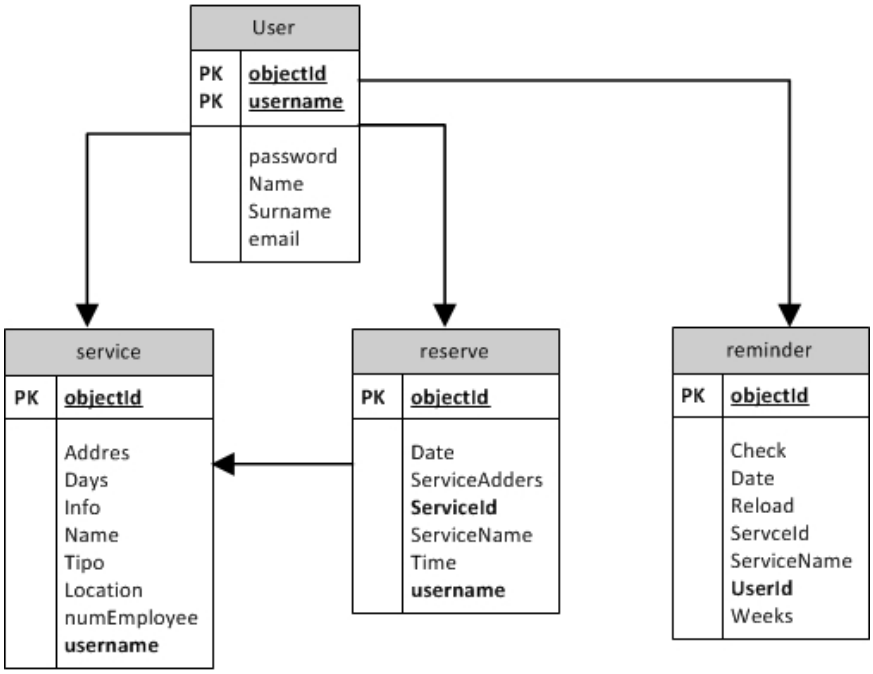
El servicio Could computing es www.parse.com.

Nos ofrece toda la información del api en:

https://www.parse.com/docs/android/api/

El servicio funciona como una base de datos, hay que crear las tablas denominadas objetos y relacionarlas, la librería nos proporciona un api para realizar consultas similares a las que podríamos realizar con SQL pero integradas en funciones del propio api.

Sistema relacional de tablas:



Como podemos ver en el gráfico el servicio quedará relacionado con el usuario por su username, las reservas quedarán relacionadas con el usuario por su username y con el servicio por el identificador objectId que lo hemos llamado ServiceId.

Finalmente el recordatorio queda relacionado con el usuario por su identificador que lo hemos llamado UserId.

El sistema de consultas de la librería Parse se basa en crear objetos de la tabla relacionada y sobre estos objetos lanzar las operaciones, podemos ver un ejemplo de cómo eliminamos una reserva:

```
ParseObject oReserve = Session.getBook();
oReserver.deleteInBackground(new DeleteCallback()
{
      @Override
      public void done(ParseException e) {


            }

      }
);
```

otros casos podemos realizar cualquier tipo de consulta con las diferentes funciones que nos proporciona la api.

Imagen web de servicio Parse donde podemos ver la tabla de servicios:

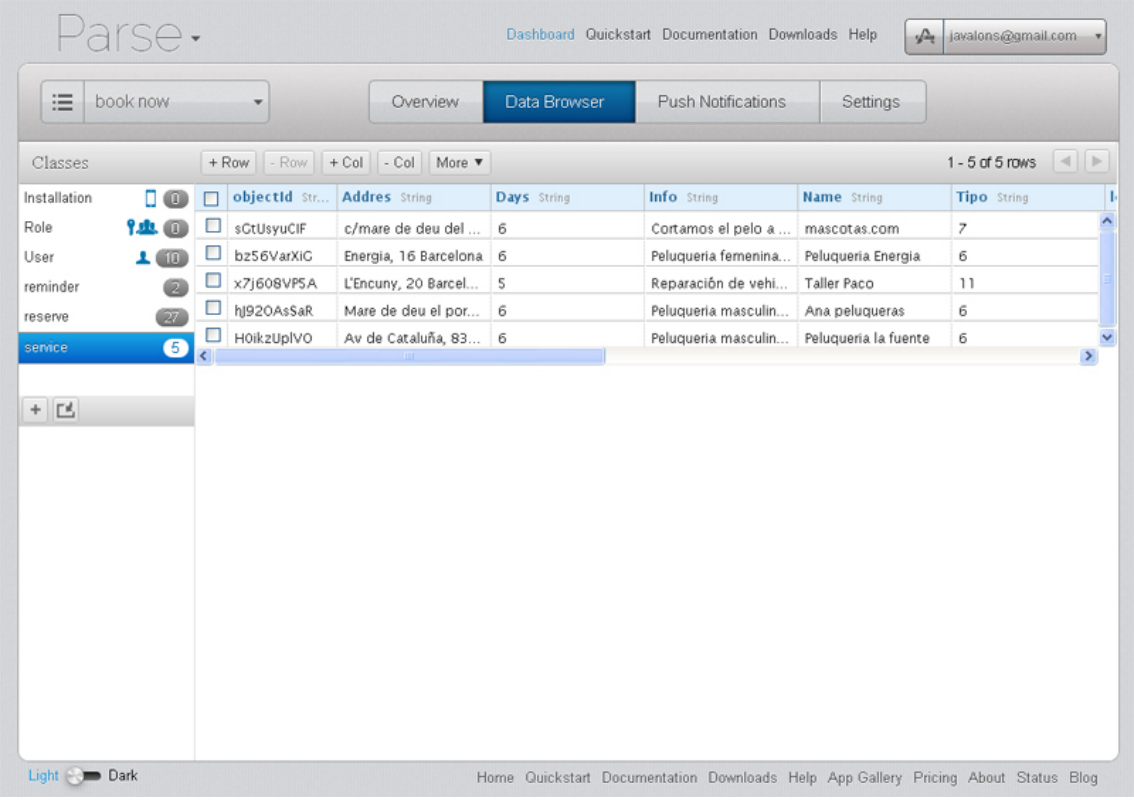


Ilustración 36. Servicio en la nube Parse

3.9 Pruebas y Test de aplicación

En este apartado se explican las diferentes pruebas y test llevados a cabo con el fin de verificar el correcto funcionamiento de la aplicación.

Muchos procesos de la aplicación requieren de la ejecución de una Activity y de respuestas recibidas por el servidor así como la interacción del usuario eso dificulta hacer pruebas de módulos unitarias por eso se han realizado pruebas unitarias por módulos de funciones muy específicas y pruebas de usabilidad para cada proceso.

Para consultar los resultados de las pruebas y test realizados consultar el Apéndice C.

Pruebas de usabilidad automáticas

En las pruebas de usabilidad se llevan a cabo las diferentes acciones que se realizan en la aplicación tanto para el lado de la empresa como del cliente y registramos los acontecimientos mediante un registro logcat de Android.

Pruebas de usabilidad manuales

Se han realizado pruebas manuales sobre la aplicación para confirmar el correcto funcionamiento sobre las diferentes acciones que se pueden realizar sobre la aplicación como puede ser todos los pasos necesarios para realizar una reserva comprobando que no se produzcan errores en los métodos que intervienen.

Pruebas de test unitarias

Se realizan pruebas unitarias sobre funciones concretas que permiten este tipo de test a través de la clase TestBookNow y guardando los resultados en el logcat de Android.

4. Resultados

4.1 Experiencias

Durante el desarrollo del proyecto he adquirido una serie de aptitudes y experiencias como el aprendizaje de nuevas tecnologías, la utilización y comprensión de Apis de terceros, la comprensión de documentación externa, la improvisación y resolución de problemas sobre la marcha, la correcta utilización de nomenclaturas e identación en el código y documentación, profundizar más sobre aspectos de programación como la optimización y la seguridad, y trabajar con un enfoque totalmente pensado en la fluidez y la simplicidad de la aplicación sin perder funcionalidad de cara a que el usuario tenga una mejor experiencia de uso de la aplicación.

Y he podido conocer la arquitectura de una aplicación Android, cómo funciona el sistema operativo a la hora de gestionar recursos y gestionar las aplicaciones, así como todas las herramientas que ofrece el compilador eclipse con el kit de desarrollo para Android.

He adquirido conocimientos de persistencia en la nube utilizando el api de desarrollo de Parse junto con servidores de datos externos.

Aunque ya tenía experiencia en la interacción con clientes en este aspecto más centrado al DCU (diseño centrado en el usuario) ha sido enriquecedor para comprobar la importancia de la colaboración de los clientes durante la fase de diseño en una aplicación móvil.

Finalmente he aprendido a gestionar el tiempo en el desarrollo y priorizar las tareas a realizar en función del tiempo.

4.2 Aportaciones propias y cambios de diseño

Durante el desarrollo del proyecto se dan una serie de complicaciones que obligan a buscar soluciones y también ves posibles cambios que mejorarían lo establecido en el primer diseño de prototipo, también se decide eliminar funcionalidades que pueden ser de menos importancia para no poner en riesgo la correcta entrega final del proyecto.

Búsquedas de servicios:

En el diseño inicial se establecen tres pantallas de búsqueda para servicios, durante la implementación se decide establecer la búsqueda en una sola pantalla donde todo queda más centralizado permitiendo desde la misma realizar la búsqueda directa de los tres modos establecidos mostrando directamente la lista de servicios permitiendo filtrar esta por nombre y con un acceso al mapa donde se mostraran los servicios relacionados.

Listas de objetos:

En la aplicación se utilizan controles del tipo ListView para mostrar la información de los diferentes objetos utilizados en la aplicación como servicios, reservas o recordatorios.

Estas listas funcionan con un objeto adaptador, estos adaptadores son sencillos y solo nos permiten mostrar un texto informativo, la idea inicial era poder mostrar 2 textos informativos y que cada objeto fuese acompañado con imágenes representativas, de ese modo para conseguir un ListView que se ajuste a nuestras necesidades creamos una clase que hereda del adaptador para sobrescribir las funciones correspondientes del objeto con el fin de transformar el objeto adaptador, es decir usamos el polimorfismo de sobrecarga para conseguir nuestro propósito.

Calendario:

Android proporciona un calendario visual del tipo CalendarView que hereda de un FrameLayout y que por defecto sus funciones no permiten editarlo, además tiene el gran inconveniente que este se incluyó a partir de la Api nivel 11 de Android de manera que solo nos permitiría utilizarlo en dispositivos que tengan una versión del sistema operativo 3.0 (HONEYCOMB) o superior, actualmente hay muchos dispositivos en el mercado inferiores a esa versión normalmente los encontramos con la versión 2.3 (GINGERBREAD) , de esta manera se decidió buscar una alternativa , la solución como ya hicimos con el objeto adaptador para los ListView es utilizar un objeto de tipo GridView y modificar su adaptador para transformarlo en un calendario sobrecargando las funciones correspondientes y añadiendo nuevas funcionalidades.

Navegación por activities:

Dentro de una aplicación Android cuando cargamos actividades nuevas la anterior que estaba abierta no se cierra a no ser que se lo indiquemos antes de indicar la nueva actividad, esto funciona de esta manera para que el dispositivo guarde en reposo la última actividad porque normalmente volveremos a ella con el botón de retroceso.

Normalmente las aplicaciones suelen tener una navegación lineal que para movernos entre pantallas o actividades pero en nuestra aplicación nos encontrábamos con una situación en la que nos interesaba una vez realizada la reserva cerrar todas las actividades abiertas hasta llegar al proceso de reserva y dejar la aplicación en la pantalla principal.

Para resolver esto se utiliza el objeto de tipo Handle para enviar mensajes a través de las actividades y se crea un servicio que gestiona los mensajes enviándolos al destinatario correspondiente según un identificador de actividad.

De esta manera cuando necesitamos cerrar una actividad en la que no nos encontramos solo tenemos que utilizar el servicio de mensajería para indicar que actividades queremos que se cierren, y cargando la actividad a la que nos queremos dirigir.

Eliminación de la funcionalidad edición de reservas:

Esta funcionalidad lo único que nos permitía era cambiar la fecha y la hora de una reserva, por causas de desarrollo ya que la idea era hacerlo de forma visual navegando por el calendario y de tiempo se decide prescindir de esta funcionalidad ya que si se quiere modificar una reserva siempre podemos eliminarla y volverla a crear siendo un proceso bastante rápido ya que se realiza en pocos pasos y sin necesidad de introducir información.

4.3 Críticas y mejoras

La aplicación tiene varios puntos en los que se puede mejorar, en este apartado intentaré hacer un crítica sobre estos puntos y explicar cómo podríamos mejorarlos.

Sincronización de agendas.

Actualmente la aplicación en modo empresa realiza una sincronización cuando esta inicia o cuando volvemos a la pantalla principal, si alguien realiza una reserva no veremos la agenda sincronizada si no salimos de la aplicación y volvemos a entrar o si realizamos alguna acción para volver a sincronizar como realizar una reserva.

Para mejorar este aspecto la aplicación en modo empresa tendría que tener un servicio que pasado un periodo de tiempo sincronizará la agenta automáticamente o una opción manual para que el usuario la sincronice.

Calendario

El calendario se ha montado utilizando un GridView, de esta manera la interface gráfica del calendario la hemos montado manualmente, pienso que se podría haber logrado una interface gráfica mejor en la que se indicarán los días de la semana no solo numéricamente sino también por su nombre , en la parte de código la estructura funcional es un poco confusa y cuesta entender cómo funciona el calendario si no lo ha programado uno mismo ese es un módulo que se puede mejorar reescribiendo el código por uno más limpio e intuitivo.

Reservas

Las reservas tienen varios puntos que se pueden mejorar , unos de ellos es el hecho de que actualmente si no se borran las reservas manualmente una vez ha pasado el día de la reserva esta se queda en el servidor , hay dos maneras de verlo por un lado puede que no sea mala opción que estén cargadas aunque ya caducadas a modo de registro pero tal vez tendría que ser un registro con un año de antigüedad si alguna reserva pasa del año se tendría que crear un servicio que las eliminará automáticamente.

Por otro lado la aplicación ahora permite realizar reservas sin un límite de fecha es decir yo podría realizar una reserva para el año 2033, se tendría que establecer un tiempo límite para realizar las reservas.

Otra opción a mejorar es que el usuario empresa pueda eliminar varias reservas a la vez.

Mapa

En el mapa cuando buscamos servicios si pulsamos sobre el servicio que deseamos actualmente se muestra un cuadro de diálogo que nos pregunta si queremos ver los detalles de ese servicio, lo ideal sería que sobre el mismo mapa se indicase ya un pequeño detalle del servicio y al pulsar sobre este detalle fuésemos a la pantalla principal.

Usuario

El usuario empresa tiene una casilla de información donde puede poner detalles sobre su empresa anunciar promociones etc. Actualmente esta casilla solo la puede modificar si se pone en contacto con el distribuidor de la aplicación para que el la modifique, este campo sería adecuado que lo pudiese editar el propio usuario desde la aplicación.

Franja horaria para las reservas

En este momento la franja horaria tiene unas horas fijas para realizar las reservas, es muy probable que unas empresas tengan horarios diferentes a otras y que sus servicios no tengan tanta duración como para establecer una hora por defecto para cada servicio, este punto se podría mejorar permitiendo al usuario empresa editar los tiempos para cada reserva pudiendo elegir cuando empieza y termina su jornada laboral y que duración tiene el servicio.

Interface gráfica

La idea inicial era que cada servicio se relacionara con un icono que lo diferenciase por ejemplo las peluquerías con un icono de unas tijeras pero como la aplicación se puede actualizar con la incorporación de nuevos servicios hay que contemplar que tendremos que buscar el icono apropiado para cada nuevo servicio. También sería un punto de ayuda al cliente que se mostraran fotografías del centro donde se realizan los servicios en la pantalla de detalles de servicio.

4.4 Conclusiones

La estrategia establecida para el desarrollo del proyecto ha tenido un resultado positivo ya que se ha finalizado el proyecto y se ha logrado cumplir con el objetivo propuesto realizando una aplicación nativa para el sistema operativo Android cumpliendo con casi la totalidad de las funcionalidades especificadas.

Una parte importante de esta estrategia fue la formación previa para entender cómo funciona y como se trabaja sobre Android.

Se han adquirido los conocimientos y experiencias que se esperan al finalizar este tipo de proyectos.

Para mí ha sido una gran experiencia desde todos los puntos, fases del proyecto y la interacción tanto con consultores como con los clientes potenciales.

Glosario

Termino	Definición
Adobe Photoshop	Software de diseño gráfico
Activity	Clase de representación y ciclo de vida de una ventana en una aplicación Android.
ADT	Pluguin de desarrollo Android para trabajar sobre el compilador Eclipse.
Android	Sistema operativo para dispositivos móviles.
ArrayList	Clase java para la utilización de listas de objetos.
Callback	Función que se ejecuta cuando se recibe una llamada a la cual espera.
Cloud Computing	Tecnología de almacenamiento de datos en un servidor remoto.
DCU	Diseño de una aplicación centrándonos en el usuario.
Eclipse	Software de desarrollo para aplicaciones.
EditText	Control java para la edición de textos.
FrameLayout	Clase java para la representación de interfaces.
GridView	Control java para mostrar datos en una rejilla o cuadrícula.
Google Apis	Librerías de Google para la utilización de sus servicios.
GPS	Sistema de posicionamiento Global que nos permite fijar la posición de un objeto en un mapa sobre unas coordenadas.
Handler	Objeto java para el proceso de eventos o manipulador.
List	Clase java para almacenar objetos en forma de lista.
Location	Clase de datos Android que representa la

	localización geográfica.
Logcat	Herramienta Android para mostrar los eventos registrados durante el proceso de la aplicación.
LoguInCallback	Función callback para el registro de usuarios.
Overlay	Sistema gráfico para la superposición de imágenes.
Parse	Servicio de persistencia de datos en la nube.
ParseObject	Objeto general de la librería Parse.
ParseUser	Objeto usuario de la librería Parse.
RadioButton	Control java modo interruptor para selección.
SDK	Kit de desarrollo de software.
SingUpCallback	Función callback para la identificación de usuarios.
SQLite	Base de datos y librería para persistencia local de datos.
Spinner	Control para la representación de menús.
View	Clase Android para la representación en pantalla de componentes.
ViewGroup	Clase para el almacenamiento de otras clases hijas View.
XML	Lenguaje de marcas extensible para la definición y transmisión de información.

Bibliografía

Android 4 Anaya y Addison Wesley (Original: Android Wireless Application Development Volume I). **Lauren Darcey y Shane Conder 2012.**

Java a fondo. Estudio del lenguaje y desarrollo de aplicaciones, edición Ra-Ma. **Pablo Augusto Sznajdlerder.**

TheNewBoston – Android Application Development

http://www.nosolousabilidad.com/articulos/dcu.htm

http://web.mit.edu/21w.789/www/papers/p55-kangas.pdf

http://www.uiaccess.com/JustAsk/es/index.html

http://developer.android.com/sdk/index.html

http://www.sgoliver.net/blog/?page_id=3011

http://stackoverflow.com/

https://www.parse.com/docs/android/api/

Apéndices.

Apéndice A. DCU, Test funcionales.

Pruebas de test sobre usuarios utilizando la plantilla definida para este propósito.

Puesta en práctica del Test con usuario potencial del perfil Cliente.

TEST PARA APLICACIÓN DE RESERVAS EN DISPOSITIVO MOVIL	
1. Usuario	
Nombre: **Raquel Pérez**	
Edad: **34**	
Sexo: **Femenino**	
Profesión o Ocupación: **Atención al cliente en un centro de servicios.**	
1.1 ¿Qué relación tienes con la tecnología, que dispositivos o medios sueles utilizar?	**Móvil, ordenador, internet.**
1.2 ¿Qué tipo de aplicaciones sueles usar?	**Mensajería, juegos, gestión.**
1.3 ¿De las aplicaciones que usas cuáles te resultan más atractivas?	**Los juegos, por ejemplo juegos de póker.**
1.4 ¿De las aplicaciones que usas cuál te resulta más fácil de utilizar?	**Las de gestión.**
1.5 ¿Dónde sueles utilizar las aplicaciones? Casa, trabajo, trasporte público etc.	**En cualquier sitio, preferentemente en casa.**
1.6 ¿Cuándo reservas hora para algún servicio como por ejemplo pedir hora para la peluquería como sueles hacerlo, por teléfono o te desplazas al centro?	**Por teléfono.**
1.7 ¿Si pudieras pedir el servicio a través de una aplicación móvil, lo utilizarías?	**Sí**
1.8 ¿El uso de una aplicación móvil de reservas te sería útil para tú profesión u ocupación?	**No**
2. Evalúa de 1 a 5 las siguientes frases	
2.1 Aplicación con muchas funcionalidades aunque te cueste más entenderla.	**1**
2.2 Aplicación con textos informativos.	**2**

2.3 Aplicación con imágenes descriptivas.	**5**
2.4 Aplicación con muchas pantallas.	**1**
2.5 Aplicación de entorno muy cargado de componentes e información.	**1**
2.6 Aplicación de entorno diáfano	**1**
3. Trabajos de usuario **Valora de 1 a 5 las siguientes frases**	
3.1 Registrarse en la aplicación como usuario aportando nombre, dirección de correo electrónico y contraseña.	**5**
3.2 Buscar un servicio para consultar o realizar una reserva.	**4**
3.3 Realizar la búsqueda por el nombre del servicio, por ejemplo Talleres Iglesias.	**5**
3.4 Realizar la búsqueda sobre un mapa cerca de donde te encuentras.	**5**
3.5 Realizar la búsqueda en una lista donde salen todos los servicios.	**1**
3.6 Realizar una reserva seleccionando el día a través del calendario.	**5**
3.7 Seleccionar la hora para la reserva con unas horas fijas y prefijadas.	**5**
3.8 Consultar fechas disponibles para reservar en un calendario.	**5**
3.9 Realizar un recordatorio que te avise para próximas reservas similares.	**5**
3.10 Realizar una consulta detallada que muestre en nombre la dirección y la hora de la reserva.	**5**
3.11 Cancelar una reserva sólo con más de una hora de antelación.	**5**
3.12 Editar una reserva sólo con más de una hora de antelación.	**5**
3.13 Consultar la cuenta de usuario.	**5**

3.14 Editar la cuenta de usuario.	**5**
3.15 Eliminar la cuenta de usuario.	**5**
4. Preguntas sobre las tareas	
4.1 ¿Te sería útil referenciar los servicios por colores, por ejemplo todos los servicios relacionados con la estética que estén identificados con el color verde?	**si**
4.2 ¿Si realizas reservas habitualmente al mismo centro, te gustaría tener ventajas y que las pudieras consultar en la aplicación?	**si**
4.3 ¿Cuándo realizas una búsqueda por localización geográfica, te gustaría saber tú posición en el mapa?	**si**
4.4 ¿Cuándo se acerque el día y la hora de la reserva te gustaría que la aplicación te avisara?	**si**
4.5 ¿Te sería de utilidad poder ordenar los resultados de las búsquedas por diferentes campos por ejemplo precio, o nombre?	**no**
4.6 ¿Te molestaría que en alguna zona discreta de la aplicación saliera publicidad relacionada con el servicio?	**si**
4.7 ¿Mientras realizas una búsqueda, te sería de utilidad que la aplicación te mostrara ofertas de servicios relacionados?	**no**
4.8 ¿Cuándo consultas los detalles de una reserva te gustaría poder ver la dirección representada en un mapa?	**si**
4.9 ¿Si eres una empresa te molestaría que los usuarios pudieran conocer los días y horas libres que tienes para que reserven?	**si**
Observaciones de usuario. **Si tienes alguna idea o mejora la puedes explicar en la siguiente casilla.**	
No se han realizado observaciones.	

Puesta en práctica del Test con usuario potencial del perfil Empresa de peluquería.

TEST PARA APLICACIÓN DE RESERVAS EN DISPOSITIVO MOVIL	
1. Usuario	
Nombre: **Alberto**	
Edad: **32**	
Sexo: **Masculino**	
Profesión o Ocupación: **Peluquero.**	
1.1 ¿Qué relación tienes con la tecnología, que dispositivos o medios sueles utilizar?	**Móvil, ordenador, internet.**
1.2 ¿Qué tipo de aplicaciones sueles usar?	**Mensajería, redes sociales, gestión.**
1.3 ¿De las aplicaciones que usas cuáles te resultan más atractivas?	**Red social Google +.**
1.4 ¿De las aplicaciones que usas cuál te resulta más fácil de utilizar?	**Mensajería.**
1.5 ¿Dónde sueles utilizar las aplicaciones? Casa, trabajo, trasporte público etc.	**En cualquier sitio, siempre que tenga buena conexión.**
1.6 ¿Cuándo reservas hora para algún servicio como por ejemplo pedir hora para la peluquería como sueles hacerlo, por teléfono o te desplazas al centro?	**Por teléfono.**
1.7 ¿Si pudieras pedir el servicio a través de una aplicación móvil, lo utilizarías?	**si**
1.8 ¿El uso de una aplicación móvil de reservas te sería útil para tú profesión u ocupación?	**si**
2. Evalúa de 1 a 5 las siguientes frases	
2.1 Aplicación con muchas funcionalidades aunque te cueste más entenderla.	**1**
2.2 Aplicación con textos informativos.	**4**
2.3 Aplicación con imágenes descriptivas.	**4**
2.4 Aplicación con muchas pantallas.	**3**

2.5 Aplicación de entorno muy cargado de componentes e información.	**1**
2.6 Aplicación de entorno diáfano	**4**
3. Trabajos de usuario **Valora de 1 a 5 las siguientes frases**	
3.1 Registrarse en la aplicación como usuario aportando nombre, dirección de correo electrónico y contraseña.	**4**
3.2 Buscar un servicio para consultar o realizar una reserva.	**4**
3.3 Realizar la búsqueda por el nombre del servicio, por ejemplo Talleres Iglesias.	**5**
3.4 Realizar la búsqueda sobre un mapa cerca de donde te encuentras.	**5**
3.5 Realizar la búsqueda en una lista donde salen todos los servicios.	**3**
3.6 Realizar una reserva seleccionando el día a través del calendario.	**5**
3.7 Seleccionar la hora para la reserva con unas horas fijas y prefijadas.	**5**
3.8 Consultar fechas disponibles para reservar en un calendario.	**5**
3.9 Realizar un recordatorio que te avise para próximas reservas similares.	**5**
3.10 Realizar una consulta detallada que muestre en nombre la dirección y la hora de la reserva.	**5**
3.11 Cancelar una reserva sólo con más de una hora de antelación.	**5**
3.12 Editar una reserva sólo con más de una hora de antelación.	**5**
3.13 Consultar la cuenta de usuario.	**3**
3.14 Editar la cuenta de usuario.	**3**
3.15 Eliminar la cuenta de usuario.	**2**

4. Preguntas sobre las tareas	
4.1 ¿Te sería útil referenciar los servicios por colores, por ejemplo todos los servicios relacionados con la estética que estén identificados con el color verde?	**si**
4.2 ¿Si realizas reservas habitualmente al mismo centro, te gustaría tener ventajas y que las pudieras consultar en la aplicación?	**si**
4.3 ¿Cuándo realizas una búsqueda por localización geográfica, te gustaría saber tú posición en el mapa?	**si**
4.4 ¿Cuándo se acerque el día y la hora de la reserva te gustaría que la aplicación te avisara?	**si**
4.5 ¿Te sería de utilidad poder ordenar los resultados de las búsquedas por diferentes campos por ejemplo precio, o nombre?	**no**
4.6 ¿Te molestaría que en alguna zona discreta de la aplicación saliera publicidad relacionada con el servicio?	**no**
4.7 ¿Mientras realizas una búsqueda, te sería de utilidad que la aplicación te mostrara ofertas de servicios relacionados?	**si**
4.8 ¿Cuándo consultas los detalles de una reserva te gustaría poder ver la dirección representada en un mapa?	**si**
4.9 ¿Si eres una empresa te molestaría que los usuarios pudieran conocer los días y horas libres que tienes para que reserven?	**no**
Observaciones de usuario. **Si tienes alguna idea o mejora la puedes explicar en la siguiente casilla.**	
Sería de utilidad poder saber el tipo de cliente que ha realizado la reserva para tener un control de la duración, no es lo mismo cortar el pelo a un niño que a un adulto o un anciano, también se tendría que tener en cuenta que hay gente que realiza reservas y después no acude al servicio poder identificar a estos usuarios para tomar las medidas correspondientes.	

Puesta en práctica del Test con usuario potencial del perfil Empresa de taller de reparación.

TEST PARA APLICACIÓN DE RESERVAS EN DISPOSITIVO MOVIL	
1. Usuario	
Nombre: **José Antonio**	
Edad: **52**	
Sexo: **Masculino**	
Profesión o Ocupación: **Mecanico.**	
1.1 ¿Qué relación tienes con la tecnología, que dispositivos o medios sueles utilizar?	**Móvil, ordenador, internet.**
1.2 ¿Qué tipo de aplicaciones sueles usar?	**Agenda, bloc de botas, gestión.**
1.3 ¿De las aplicaciones que usas cuáles te resultan más atractivas?	**Agenda.**
1.4 ¿De las aplicaciones que usas cuál te resulta más fácil de utilizar?	**Agenda.**
1.5 ¿Dónde sueles utilizar las aplicaciones? Casa, trabajo, trasporte público etc.	**En el trabajo.**
1.6 ¿Cuándo reservas hora para algún servicio como por ejemplo pedir hora para la peluquería como sueles hacerlo, por teléfono o te desplazas al centro?	**Por teléfono.**
1.7 ¿Si pudieras pedir el servicio a través de una aplicación móvil, lo utilizarías?	**si**
1.8 ¿El uso de una aplicación móvil de reservas te sería útil para tú profesión u ocupación?	**si**
2. Evalúa de 1 a 5 las siguientes frases	
2.1 Aplicación con muchas funcionalidades aunque te cueste más entenderla.	**2**
2.2 Aplicación con textos informativos.	**3**
2.3 Aplicación con imágenes descriptivas.	**3**
2.4 Aplicación con muchas pantallas.	**4**

2.5 Aplicación de entorno muy cargado de componentes e información.	**2**
2.6 Aplicación de entorno diáfano	**5**
3. Trabajos de usuario **Valora de 1 a 5 las siguientes frases**	
3.1 Registrarse en la aplicación como usuario aportando nombre, dirección de correo electrónico y contraseña.	**4**
3.2 Buscar un servicio para consultar o realizar una reserva.	**5**
3.3 Realizar la búsqueda por el nombre del servicio, por ejemplo Talleres Iglesias.	**4**
3.4 Realizar la búsqueda sobre un mapa cerca de donde te encuentras.	**3**
3.5 Realizar la búsqueda en una lista donde salen todos los servicios.	**3**
3.6 Realizar una reserva seleccionando el día a través del calendario.	**4**
3.7 Seleccionar la hora para la reserva con unas horas fijas y prefijadas.	**5**
3.8 Consultar fechas disponibles para reservar en un calendario.	**5**
3.9 Realizar un recordatorio que te avise para próximas reservas similares.	**5**
3.10 Realizar una consulta detallada que muestre en nombre la dirección y la hora de la reserva.	**5**
3.11 Cancelar una reserva sólo con más de una hora de antelación.	**5**
3.12 Editar una reserva sólo con más de una hora de antelación.	**5**
3.13 Consultar la cuenta de usuario.	**5**
3.14 Editar la cuenta de usuario.	**3**
3.15 Eliminar la cuenta de usuario.	**3**

4. Preguntas sobre las tareas	
4.1 ¿Te sería útil referenciar los servicios por colores, por ejemplo todos los servicios relacionados con la estética que estén identificados con el color verde?	**no**
4.2 ¿Si realizas reservas habitualmente al mismo centro, te gustaría tener ventajas y que las pudieras consultar en la aplicación?	**si**
4.3 ¿Cuándo realizas una búsqueda por localización geográfica, te gustaría saber tú posición en el mapa?	**si**
4.4 ¿Cuándo se acerque el día y la hora de la reserva te gustaría que la aplicación te avisara?	**si**
4.5 ¿Te sería de utilidad poder ordenar los resultados de las búsquedas por diferentes campos por ejemplo precio, o nombre?	**si**
4.6 ¿Te molestaría que en alguna zona discreta de la aplicación saliera publicidad relacionada con el servicio?	**no**
4.7 ¿Mientras realizas una búsqueda, te sería de utilidad que la aplicación te mostrara ofertas de servicios relacionados?	**si**
4.8 ¿Cuándo consultas los detalles de una reserva te gustaría poder ver la dirección representada en un mapa?	**si**
4.9 ¿Si eres una empresa te molestaría que los usuarios pudieran conocer los días y horas libres que tienes para que reserven?	**si**
Observaciones de usuario. **Si tienes alguna idea o mejora la puedes explicar en la siguiente casilla.**	
Me gustaría poder mostrar ofertas sobre promociones como cambios de aceite y de neumáticos y que los usuarios las pudieran ver.	

Apéndice B. Implementación, Referencias de métodos.

En este apéndice se documentan los principales métodos de los módulos que forman la aplicación.

Activity

Directorio de archivos **src.com.tfc.Activities**.

Actividad Inicial

Activity Init	
Archivo	init.java
Layout	init.xml
Relaciones	Actividad de inicio de sesión, **SinginUp.java**. Actividad principal en modo cliente, **MainActivity.java**. Actividad principal en modo empresa, **MainCompanyActivity.java**. Módulo de sesión de usuario, **Session.java**. Driver de base de datos SQLite, **DriverDBSQLite.java**.

Actividad de Inicio de sesión

Activity SinginUp	
Archivo	SinginUp.java.
Layout	Activity_singin_up.xml.
Relaciones	Actividad de registro de usuario, **Register.java**. Actividad principal en modo cliente, **MainActivity.java**. Actividad principal en modo empresa, **MainCompanyActivity.java**. Módulo de sesión de usuario, **Session.java**.

Funciones principales

public static void logInInBackground(String username,String password,LogInCallback callback)	
Parámetros	Usuario - mail del usuario que inicia la sesión Contraseña - contraseña del usuario Callback - callback que se llamará cuando finalice la operación.
Respuesta	Respuesta tipo Callback. ParseException, ParseUser (Objeto Usuario)
Objetivo	Inicializa la sesión de usuario, la respuesta se recibe a través de un callback, si la respuesta es positiva se reciben los datos del usuario a través de un objeto ParseUser.

Actividad de registro de usuario

Activity Register	
Archivo	Register.java.
Layout	Activity_register.xml.
Relaciones	Actividad principal en modo cliente, **MainActivity.java**. Módulo de sesión de usuario, **Session.java**.

Funciones principales

public void **signUpInBackground**(SignUpCallback callback)	
Parámetros	Callback - callback que se llamará cuando finalice la operación.
Respuesta	Respuesta tipo Callback. ParseException - Lanza una excepción si no se puede conectar con el servidor, o si el nombre de usuario ya existe.
Objetivo	Guarda un nuevo usuario en el servidor.

Actividad principal de usuario cliente

Activity Main	
Archivo	MainActivity.java.
Layout	Activity_main.xml.
Relaciones	Actividad de lista de servicios, **ListServicesActivity.java**. Actividad de detalle de reserva, **DetailReserveActivity.java**. Actividad de inicio de sesión, **SinginUp.java.** Actividad de mapa. **SelectedActivity.java.** Clase usuario, **User.java**. Módulo de constantes, **Constants.java**. Módulo de sesión de usuario, **Session.java**. Módulo de mensajería, **activityMessage.java**. Módulo de mensajes de diálogo, **dialogMessage.java**. Ítem de control, **ItemReserveAdapter.java**. Ítem de control, **ItemServices.java**

Funciones principales

public void **ShowServices**()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Carga los datos de los tipos de servicios y los muestra en pantalla.

public void **LoadBooks**()

Parámetros	No tiene parámetros.
Respuesta	Respuesta tipo Callback. Lista de usuarios en modo de objetos ParseUser. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la consulta.
Objetivo	Realiza una consulta al servidor para obtener todas las reservas relacionadas con el usuario.

public void **ShowBooks**()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Carga todos los datos de las reservas para mostrarlos en pantalla.

public void **ShowInfoUser**()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Recupera los datos de usuario de la sesión iniciada y los carga para mostrar en pantalla.

public void **CheckAndUpdateUser**()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta tipo mensaje Handle. Se recibe una respuesta a través del objeto updateMessage de tipo Handle creado dentro de la actividad principal. Se espera el mensaje MSG_UPDATE_DATA_USER.
Objetivo	Chequea los datos de usuario si son correctos llama a la función EditUserData del objeto Usuario para editar sus datos.

Actividad principal de usuario empresa

Activity CompanyMain	
Archivo	CompanyMainActivity.java.
Layout	Activity_company_main.xml.
Relaciones	Actividad de detalle de reserva, **DetailReserveActivity.java**. Actividad de inicio de sesión, **SinginUp.java.** Clase usuario, **User.java**. Clase reserva, **Reserve.java**.

	Módulo de útiles, **Tools.java**. Módulo de constantes, **Constants.java**. Módulo de sesión de usuario, **Session.java**. Módulo de mensajería, **activityMessage.java**. Ítem de control, **ItemReserveAdapter.java**. Ítem de control, **ItemServices.java.** Ítem de control, **MyCalendarAdapter.**

Funciones principales

public void confirmReserve(String sName,int iTime)	
Parámetros	sName - Nombre de usuario para la reserva. iTime - Horario de la reserva.
Respuesta	Respuesta tipo mensaje Handle. Se recibe una respuesta a través del objeto updateMessage de tipo Handle creado dentro de la actividad principal. Se espera el mensaje MSG_SAVE_BOOK_OK.
Objetivo	Prepara los datos para la reserva y llama a la función SaveReserve del objeto Reserve.

public void getServiceUser()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta tipo Callback. Lista de servicios en modo de objetos ParseObject. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la consulta.
Objetivo	Recupera el servicio relacionado con el usuario.

public void getBooksAndShowInfo()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta tipo Callback. Lista de reservas en modo de objetos ParseObject. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la consulta.
Objetivo	Recupera las reservas relacionado con el usuario y llama a la funciones ShowBooks y ShowCalendar para representar la información.

public void ShowBooks(int iDay)	
Parámetros	iDay - Dia para el que queremos mostrar las reservas.
Respuesta	No tiene respuesta.

Objetivo	Muestra en pantalla las reservas del día correspondiente.

public void ShowCalendar()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Prepara los datos del calendario para poder representarlo en pantalla, se encarga de llamar a la función prepareCalendar del adaptador del calendario para editar los gráficos del calendario si un día tiene una reserva o está completo lo mostrará con el grafico correspondiente.

public void ShowInfoUser()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Recupera los datos de usuario de la sesión iniciada y los carga para mostrar en pantalla.

public void CheckAndUpdateUser()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta tipo mensaje Handle. Se recibe una respuesta a través del objeto updateMessage de tipo Handle creado dentro de la actividad principal. Se espera el mensaje MSG_UPDATE_DATA_USER.
Objetivo	Chequea los datos de usuario si son correctos llama a la función EditUserData del objeto Usuario para editar sus datos.

Actividad de lista de servicios

Activity ListService	
Archivo	ListServiceActivity.java.
Layout	Activity_list_service.xml.
Relaciones	Actividad de detalle de servicio, **DetailServiceActivity.java**. Actividad de localización geográfica, **SelectedActivity.java.** Módulo de sesión de usuario, **Session.java**. Módulo de mensajería, **activityMessage.java**. Módulo de mensajes de diálogo, **dialogMessage.java**. Ítem de control, **ItemReserveAdapter.java**. Ítem de control, **ItemServices.java.**

Funciones principales

public void NameFilter()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Filtra los servicios por nombre y los carga para mostrar en pantalla.

public void ShowSearchService()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta tipo Callback. Lista de servicios en modo de objetos ParseObject. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la consulta.
Objetivo	Recuperar todos los servicios del tipo relacionado y mostrarlos en pantalla.

Actividad de detalla de servicio

Activity DetailService	
Archivo	DetailServiceActivity.java.
Layout	Activity_detail_service.xml.
Relaciones	Actividad de calendario, **CalendarActivity.java**. Módulo de sesión de usuario, **Session.java**. Módulo de mensajería, **activityMessage.java**.

Funciones principales

public void ShowInfo()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Carga en los controles del layout la información del servicio para mostrarla en pantalla.

Actividad de detallada de reserva

Activity DetailReserve	
Archivo	DetailReserveActivity.java.
Layout	Activity_detail_reserve.xml.
Relaciones	Actividad de localización geográfica, **SelectedActivity.java.** Clase usuario, **User.java**. Clase reserva, **Reserve.java**. Clase recordatorio, **Reminder.java**. Módulo de útiles, **Tools.java**. Módulo de constantes, **Constants.java**. Módulo de sesión de usuario, **Session.java**. Módulo de mensajería, **activityMessage.java**.

Funciones principales

public void **ShowInfo**()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Carga en los controles del layout la información de la reserva para mostrarla en pantalla.

public void **SearchService**()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta tipo Callback. Lista de servicios en modo de objetos ParseObject. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la consulta.
Objetivo	Recuperar el servicio relacionado con la reserva y lanza la actividad de localización para mostrarlo en el mapa.

void **confirmReminder**()	
Parámetros	ldItemSelected - id de control “radiobutton” seleccionado. bReload - información de repetición tipo booleano.
Respuesta	Respuesta tipo mensaje Handle. Se recibe una respuesta a través del objeto updateMessage de tipo Handle creado dentro de la actividad principal. Se espera el mensaje MSG_SAVE_REMINDER_OK.
Objetivo	Llamar a la función CheckExistReminder para comprobar si tenemos ya recordatorios con ese servicio, crear un objeto de recordatorio (Reminder) y lanzar su función SaveReminder para guardar el recordatorio en el servidor.

boolean confirmReminder()	
Parámetros	sServiceId – Identificador del servicio a comprobar.
Respuesta	Respuesta tipo booleana, retorna true si no hay ningún recordatorio creado y false si encuentra algún recordatorio.
Objetivo	Comprobar que no hay ningún recordatorio creado para el servicio seleccionado.

Actividad de calendario

Activity Calendar	
Archivo	CalendarActivity.java.
Layout	Activity_calendar.xml.
Relaciones	Clase reserva, **Reserve.java**. Módulo de útiles, **Tools.java**. Módulo de sesión de usuario, **Session.java**. Módulo de mensajería, **activityMessage.java**. Módulo de mensajes de diálogo, **dialogMessage.java**. Ítem de control, **MyCalendarAdapter.java.**

Funciones principales

void CloseActivities()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuestas.
Objetivo	Enviar un mensaje a través del módulo de mensajería activityMessage para cerrar las actividades abiertas con anterioridad e informar a la actividad principal que vuelva a cargar la lista de reservas.

boolean CheckCloseDay()	
Parámetros	iDay – día que queremos comprobar.
Respuesta	Respuesta tipo booleana, nos retorna true si el true si ese día el servicio abre y false si no abre.
Objetivo	Comprobar si el día pasado por parámetro el servicio abre, los servicios pueden abrir 5 o 6 días, si abren 5 se comprueba que no sea sábado ni domingo, si abren 6 días se comprueba que no sea domingo.

public void confirmReserve()	
Parámetros	iTime – Hora de la reserva.
Respuesta	Respuesta tipo mensaje Handle. Se recibe una respuesta a través del objeto updateMessage de tipo Handle creado dentro de la actividad principal. Se espera el mensaje MSG_SAVE_BOOK_OK.
Objetivo	Preparar los datos de la reserva para crear un objeto reserva (Reserve) y lanzar su función SaveReserve para guardar la reserva en el servidor.

public void LoadBooks()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta tipo Callback. Lista de reservas en modo de objetos ParseObject. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la consulta.
Objetivo	Realiza una consulta al servidor para obtener todas las reservas relacionadas con el usuario sobre ese servicio.

public void refreshCalendar()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Refresca los datos des calendario según el mes que tengamos cargado en el adaptador (MyCalendarAdapter).

Actividad de selección geográfica

Activity Selected	
Archivo	SelectedActivity.java.
Layout	Activity_selected.xml.
Relaciones	Módulo de sesión de usuario, **Session.java**. Módulo de constantes, **Constants.java**. Módulo de mensajería, **activityMessage.java**. Ítem de control gps, **Geo.java.** Ítem de control puntos gps, **MyPoint.java.** Ítem de control overlay, **MyOverlayItem.java.**

Funciones principales

public void **onLocationChanged**()	
Parámetros	newLocation – nueva localización del dispositivo.
Respuesta	No tiene respuesta.
Objetivo	Recupera la nueva posición del dispositivo, guarda los datos para la posición del usuario y lo representa en el mapa a través de la función DrawPoint.

public void **DrawPoint**()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Selecciona las diferentes opciones de representación de puntos geográficos en función de variables globales cargadas en la actividad y de si el GPS del dispositivo esta activado.

public void **searchService**(int iQuerry)	
Parámetros	iQuerry – tipo de consulta a realizar al servidor.
Respuesta	Respuesta tipo Callback. Lista de servicios en modo de objetos ParseObject. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la consulta.
Objetivo	Busca los servicios de el sector seleccionado, si además se le pasa a través del parámetro iQuerry el valor GEO_BY_AREA busca solo los servicios que se encuentran en 20km de radio con referencia a la localización del dispositivo.

public void **AddMapUserPoint**(boolean bAnimate)	
Parámetros	bAnimate – variable para confirmar si queremos centrar el mapa en el punto.
Respuesta	No tiene respuesta.
Objetivo	Añade un nuevo punto en el mapa para la localización del usuario.

public void **AddMapServicePoint**(ParseObject pService)	
Parámetros	pService – objeto de tipo ParseObject con los datos del servicio
Respuesta	No tiene respuesta.
Objetivo	Añade un nuevo punto en el mapa para la localización del servicio.

public void AddMapServicesPoints()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Añade nuevos puntos de servicios en el mapa.

private void createDialogService(ParseObject pService)	
Parámetros	pService – objeto de tipo ParseObject con los datos del servicio
Respuesta	No tiene respuesta.
Objetivo	Crea el diálogo para preguntar si se desea ver la información del servicio.

private void createDialogGps()	
Parámetros	No tiene parámetro.
Respuesta	No tiene respuesta.
Objetivo	Crea el diálogo para preguntar si se desea activar el GPS del dispositivo y marca la pregunta como realizada para no volver a crearlo.

Beans

Directorio de archivos **src.com.tfc.Beans**.

Objeto Reminder

Objeto Reminder	
Archivo	Reminder.java.
Relaciones	Módulo de sesión de usuario, **Session.java**. Módulo de constantes, **Constants.java**. Módulo de mensajería, **activityMessage.java**.
Variables	**String** ServiceId –Identificador del servicio. **String** UserId – Identificador del usuario. **String** ServiceName – Nombre del servicio. **String** Date – Fecha de inicio. **boolean** bCheck – Marca de chequeo. **boolean** bReload – Indicador para saber si es periódica. **int** iWeeks - Número de semanas para el recordatorio.

Funciones principales

static public void DeleteReminder(final Activity activity,ParseObject oReminder)	
Parámetros	activity – actividad actual. oReminder – objeto recordatorio a eliminar
Respuesta	Respuesta tipo Callback. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la operación.
Objetivo	Manda el comando para pedir al servidor que elimine el recordatorio.

static public void UpdateReminder(ParseObject oReminder, boolean bCheck)	
Parámetros	oReminder – objeto recordatorio a actualizar bCheck – opción de chequeo según el tipo de actualización
Respuesta	Respuesta tipo Callback. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la operación.
Objetivo	Manda el comando para pedir al servidor que actualice el recordatorio.

public void SaveReminder(final Activity activity)	
Parámetros	activity – actividad actual.
Respuesta	Respuesta tipo Callback. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la operación.
Objetivo	Manda el comando para pedir al servidor que guarde el recordatorio.

static public void LoadReminders(final Activity activity)	
Parámetros	activity – actividad actual.
Respuesta	Respuesta tipo Callback. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la operación.
Objetivo	Manda el comando para pedir al servidor que nos devuelva la lista de recordatorios del usuario actual.

Objeto Reserve

Objeto Reserve	
Archivo	Reserve.java.
Relaciones	Módulo de sesión de usuario, **Session.java**. Módulo de Utilidades, **Tools.java**. Módulo de mensajería, **activityMessage.java**.
Variables	**String** userName–Nombre de usuario. **String** ServiceId – Identificador del servicio. **String** Time – Franja horaria del servicio. **String** Date – Fecha del servicio. **String** serviceName – Nombre del servicio. **String** serviceAddres – Dirección del servicio. **String** objected – Identificador del objeto

Este objeto guarda los datos temporales de las reservas en uso por la aplicación.

Funciones principales

static public void DeleteReserve(final Activity activity)	
Parámetros	activity – actividad actual.
Respuesta	Respuesta tipo Callback. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la operación.
Objetivo	Manda el comando para pedir al servidor que elimine la reserva.

static public void SaveReserve(final Activity activity)	
Parámetros	activity – actividad actual.
Respuesta	Respuesta tipo Callback. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la operación.
Objetivo	Manda el comando para pedir al servidor que guarde la reserva.

Objeto User

Objeto User	
Archivo	User.java.
Relaciones	Módulo de sesión de usuario, **Session.java**. Módulo de mensajería, **activityMessage.java**.
Variables	**String** userName – Loguin de usuario. **String** Name – Nombre de usuario.

	String surName – Apellido de usuario. **String** Email – Email de usuario. **String** Type – Tipo de usuario cliente o empresa. **String** Password – Contraseña de usuario. **int** iSession – Marca de sesión abierta. **String** ObjectId – Identificador del objeto

Funciones principales

public void EditUserData(final Activity activity,final String sUser, final String sName,final String sSurname,final String sEmail,final String sPassword)	
Parámetros	activity – actividad actual. sUser – Login de usuario sName – Nombre de usuario sSurname- Apellido de usuario sEmail – Email de usuario sPassword – Contraseña de usuario.
Respuesta	Respuesta tipo Callback. ParseException - Lanza una excepción si no se puede conectar con el servidor, o no puede resolver la operación.
Objetivo	Manda el comando para pedir al servidor que actualice los datos del usuario.

Commons

Directorio de archivos **src.com.tfc.Commons**.

Clase común activityMessage

Clase activityMessage	
Archivo	activityMessage.java.
Relaciones	No tiene relaciones.

Funciones principales

public static void setMessage(Handler newUpMessage, int iId)	
Parámetros	newUpMessage – Handler creado para mensaje de la actividad. iId – identificador de la actividad.
Respuesta	No tiene respuesta.
Objetivo	Añade en nuevo handler a la lista con el id correspondiente.

public static Handler getMessage(int iId)	
Parámetros	iId – identificador de la actividad.

Respuesta	Handler relacionado con el identificador.
Objetivo	Buscar en la lista de handler en que hace referencia al identificador y retornarlo.

public static void sendMessage(int iMessage,int iAtrinb,int iId)	
Parámetros	iMessage – tipo de mensaje. iAtrib – atributos para enviar con el mensaje. iId – identificador de la actividad.
Respuesta	No tiene respuesta.
Objetivo	Buscar el handler relacionado con el identificador y mandar el tipo de mensaje que se ha pasado por parámetro.

Clase común Constants

Clase Constants	
Archivo	Constatns.java.
Relaciones	No tiene relaciones.

Esta clase guarda las variables constantes de uso común en el proyecto, esta clase no tiene ninguna función.

Clase común dialogMessage

Clase dialogMessage	
Archivo	dialogMessage.java.
Relaciones	No tiene relaciones.

Esta clase crea un diálogo informativo para mostrar información al usuario sobre algún evento producido en la aplicación.

Funciones

static public Dialog createDialogRedFail(final Activity activity)	
Parámetros	Activity – referencia de la actividad que llama a la función
Respuesta	Retorna el diálogo creado.
Objetivo	Crear un diálogo que nos muestra información sobre el estado de la red.

Clase común driverDBSQLite

Clase Constants	
Archivo	driverDBSQLite.java.
Relaciones	Clase usuario, **User.java**.

Esta clase es el driver de comunicación con la base de datos SQLite , se utiliza para realizar las consultas sobre la base de datos.

Funciones principales

public boolean **openForWrite**()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta de tipo booleano, retorna true si la operación ha sido correcta de lo contrario retorna false.
Objetivo	Abrir la base de datos en modo de escritura.

public boolean **openForRead**()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta de tipo booleano, retorna true si la operación ha sido correcta de lo contrario retorna false.
Objetivo	Abrir la base de datos en modo de lectura.

public void **closeDB**()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Cierra la base de datos.

public boolean **DeleteUserTable**()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta de tipo booleano, retorna true si la operación ha sido correcta de lo contrario retorna false.
Objetivo	Elimina los datos de la tabla de usuario.

public boolean updateSession(int session)	
Parámetros	session – variable que indica si la sesión está abierta o cerrada.
Respuesta	Respuesta de tipo booleano, retorna true si la operación ha sido correcta de lo contrario retorna false.
Objetivo	Actualiza el campo “session” de la tabla de usuario.

public boolean SaveUser(User user)	
Parámetros	user – referencia de objeto usuario.
Respuesta	Respuesta de tipo booleano, retorna true si la operación ha sido correcta de lo contrario retorna false.
Objetivo	Guarda los datos de usuario en la tabla usuario.

public User LoadUser()	
Parámetros	No tiene parámetros
Respuesta	Respuesta el usuario cargado de la base de datos.
Objetivo	Cargar el usuario situado en el primer registro de la tabla usuario.

Clase común Session

Clase Session	
Archivo	Session.java.
Relaciones	Clase usuario, **User.java**.

Esta clase estática se encarga de guardar la referencias de la sesión iniciada por el usuario, guarda una referencia al objeto de la clase usuario y las referencias temporales de los diferentes objetos que se utilicen en la sesión como servicios y reservas para poder ser consultadas desde cualquier módulo del proyecto.

Funciones principales

public static boolean NewUser(ParseUser newUsuario)	
Parámetros	newUsuario – Objeto usuario de la clase ParseObject.
Respuesta	Respuesta de tipo booleano, retorna true si la operación ha sido correcta de lo contrario retorna false.
Objetivo	Introduce los nuevos datos de sesión de usuario en la base de datos.

public static boolean checkUser()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta de tipo booleano, retorna true si la operación ha sido correcta de lo contrario retorna false.
Objetivo	Comprueba que hay un usuario guardado en la base de datos.

public static boolean checkSession()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta de tipo booleano, retorna true si la operación ha sido correcta de lo contrario retorna false.
Objetivo	Comprueba que el último usuario que usó la aplicación dejó la sesión abierta.

Clase común Tools

Clase Tools	
Archivo	Tools.java.
Relaciones	No tiene relaciones.

Funciones principales

public static boolean compareTime(int iTime)	
Parámetros	iTime – variable de franja horaria de la reserva
Respuesta	Respuesta de tipo booleano, retorna true si la franja horaria es superior a la hora actual de lo contrario retorna false.
Objetivo	Comprueba que la hora actual es superior a la hora de la franja horaria pasada por parámetro.

public static int compareData(Calendar c)	
Parámetros	c – variable de fecha a comparar
Respuesta	Respuesta de tipo enero, retorna 1 si la fecha es superior, 0 si la fecha es igual y -1 si la fecha es inferior.
Objetivo	Comprobar la fecha actual con la fecha que se le pasa por parámetro y retorna un valor en función de si es mayor, igual o menor.

public static int CalculateTotalDays(Calendar c)

Parámetros	c – variable de fecha a calcular
Respuesta	Respuesta de tipo enero, retorna los días totales de la fecha.
Objetivo	Calcula los días totales de una fecha desde el año 0 hasta la fecha pasada por parámetro.

Clase común Geo

Clase Geo	
Archivo	Geo.java.
Relaciones	No tiene relaciones.

Funciones principales

public void Init()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Carga el servicio de localización.

public Location GetLocation()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta de tipo Location, retorna un objeto Location con la última posición conocida por el dispositivo.
Objetivo	Recoge los proveedores de búsqueda por el criterio del mejor y localiza la última posición conocida del dispositivo.

public boolean GetGpsEnable()	
Parámetros	No tiene parámetros.
Respuesta	Respuesta de tipo boolean, retorna true si el GPS del dispositivo esta activa o false si esta desactivado.
Objetivo	Comprueba que el GPS del dispositivo esta activado.

public void LocationUpdate()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.

Objetivo	Configura los parámetros de actualización del localizador.

public void RemoveUpdate()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Detiene la actualización de posicionamiento del localizador.

Items

Directorio de archivos **src.com.tfc.Items**.

Clase item ItemReserveAdapter

Clase ItemReserveAdapter	
Archivo	ItemReserveAdapter.java.
Relaciones	No tiene relaciones.

Funciones Principales

public View getView(int position,View convertView,ViewGroup parent)	
Parámetros	Position- la posición del elemento dentro del adaptador. convertView – View utilizado que vamos a modificar. Parent – Objeto padre de la familia View
Respuesta	Retornamos el objeto View modificado.
Objetivo	Modificar el contenido con los nuevos parámetros cargados en el Item(ServiceItem) de la posición.

Clase item ItemService

Clase ItemService	
Archivo	ItemService.java.
Relaciones	No tiene relaciones.

Clase item MyCalendarAdapter

Clase MyCalendarAdapter	
Archivo	MyCalendarAdapter.java.
Relaciones	Clase usuario, **User.java**. Módulo de constantes, **Constants.java**.

	Módulo de sesión de usuario, **Session.java**.

Funciones principales

public View **getView**(int position,View convertView,ViewGroup parent)	
Parámetros	Position- la posición del elemento dentro del adaptador. convertView – View utilizado que vamos a modificar. Parent – Objeto padre de la familia View
Respuesta	Retornamos el objeto View modificado.
Objetivo	Crea una nueva vista para cada elemento según la posición

public int **getDayofWeek**(int iDay)	
Parámetros	iDay – día de la semana.
Respuesta	Retorna la clave SATURDAY, SUNDAY o 0.
Objetivo	Calcula si el día de la semana es sábado o domingo.

public AlertDialog.builder **createDialogTime**(int iDay)	
Parámetros	iDay – día de la semana.
Respuesta	Retorna el diálogo creado.
Objetivo	Crea un cuadro de diálogo que nos permite seleccionar la franja horaria.

public void **prepareCalendar**(final Calendar myCalendar, List<PasreObject> booklist)	
Parámetros	myCalendar – calendario de referencia para preparar el calendario. booklist – lista de reservas para ese mes.
Respuesta	No tiene respuesta.
Objetivo	Prepara las variables que de control que permiten después representar el calendario a la función getView.

public void **checkNumBook**(ArrayList<cBook> auxList, int index,int employee)	
Parámetros	auxList– lista de reservas para ese mes. Index – número de reserva actual. employee – número de empleados.
Respuesta	Respuesta de tipo booleano, retorna true si el horario esta libre y false si esta completo.

Objetivo	Comprueba si una empresa tiene una hora completa según el número de empleados.

Clase item MyOverlayItem

Clase MyOverlayItem	
Archivo	MyOverlayItem.java.
Relaciones	No tiene relaciones.

Clase item MyPoint

Clase MyPoint	
Archivo	MyPoint.java.
Relaciones	Módulo de mensajería, **activityMessage.java**.

Funciones principales

public MyPoint(Drawable m, Context context, int newType)	
Parámetros	m– Gráfico a representar. context – contexto actual de actividad. newType – tipo de punto a mostrar (POINT_USER o POINT_COMPANY).
Respuesta	No tiene respuesta.
Objetivo	Constructor de la clase MyPoint, crea el objeto con los datos pasados por parámetro.

protected boolean onTab(int index)	
Parámetros	Index – índice del ítem pulsado en el mapa
Respuesta	Respuesta de tipo booleana, retorna true.
Objetivo	Función que se activa cuando se pulsa sobre un ítem en pantalla, si el tipo de ítem es POINT_COMPANY envía un mensaje para informar.

public void insertMyPoint(MyOverlayItem item)	
Parámetros	Ítem- objeto del tipo MyOverlayItem
Respuesta	No tiene respuesta.
Objetivo	Carga el objeto pasado por parámetro en la lista global.

Services

Directorio de archivos **src.com.tfc.Services**.

Clase de servicios NotificatiónService

Clase NotificationService	
Archivo	NotificatiónService.java.
Relaciones	Actividad inicial, **Init.java.** Clase usuario, **User.java**. Clase recordatorio, **Reminder.java**. Módulo de útiles, **Tools.java**. Módulo de sesión de usuario, **Session.java**. Driver de base de datos SQLite, **DriverDBSQLite.java**.

Funciones principales

protected void **ControlTimer**()	
Parámetros	No tiene parámetros.
Respuesta	No tiene respuesta.
Objetivo	Crea un timer que ejecuta una tarea interna periódicamente.

private void **CheckReminder** (List<ParseObject> oReminderList)	
Parámetros	oReminderList – lista de objetos de recordatorio
Respuesta	No tiene respuesta.
Objetivo	Comprueba los recordatorios y si ha pasado el tiempo para recordarlo lanza una notificación.

public void **playNotification**(Context context, Class<?> cls, String textNotification, String titleNotification, int drawable)	
Parámetros	Context – contexto de aplicación Cls – Clase que se iniciará textNotification- texto que se muestra en la notificación titleNotification – título de la notificación drawable – icono de la notificación
Respuesta	No tiene respuesta.
Objetivo	Ejecuta la notificación para avisar de que hay un recordatorio.

Clase de servicios BootServiceManager

Clase BootServiceManager	
Archivo	BootServiceManager.java.
Relaciones	No tiene relaciones.

Apéndice C. Implementación, Resultados de test de aplicación.

Resultados de pruebas de usabilidad automáticas

Acciones de usuario empresa

Acción de inicio de sesión

12-09 17:52:07.708: I/BookNowLog(23590): Activity Init iniciada

12-09 17:52:09.938: I/BookNowLog(23590): DB - Se ha abierto la base de datos para lectura

12-09 17:52:09.958: I/BookNowLog(23590): DB - Se ha cargado el usuario de la base de datos

12-09 17:52:09.988: I/BookNowLog(23590): SS - Existe una sesión de usuario

12-09 17:52:10.168: I/BookNowLog(23590): Activity SinginUp iniciada

12-09 17:52:35.438: I/BookNowLog(23590): SN - Se ha iniciado sesión correctamente

12-09 17:52:35.453: I/BookNowLog(23590): DB - Se ha abierto la base de datos para escritura

12-09 17:52:35.513: I/BookNowLog(23590): DB - Se ha eliminado el usuario de la base de datos

12-09 17:52:35.573: I/BookNowLog(23590): DB - Se ha abierto la base de datos para escritura

12-09 17:52:35.598: I/BookNowLog(23590): DB - Se ha guardado el usuario de la base de datos

12-09 17:52:36.073: I/BookNowLog(23590): Activity CompanyMain iniciada

12-09 17:52:39.028: I/BookNowLog(23590): SV - Servicios cargados correctamente

12-09 17:52:42.633: I/BookNowLog(23590): RV - Reservas cargadas correctamente

Acción de eliminar reservas

12-09 17:54:10.568: I/BookNowLog(23590): Activity DetailReserve iniciada

12-09 17:55:03.448: I/BookNowLog(23590): RV - Reserva eliminada correctamente

12-09 17:55:07.878: I/BookNowLog(23590): RV - Reservas cargadas correctamente

Acción de realizar reservas

12-09 17:56:46.388: I/BookNowLog(23590): RV - Reserva grabada correctamente

12-09 17:57:13.323: I/BookNowLog(23590): RV - Reservas cargadas correctamente

Acción de editar cuenta de usuario

12-09 17:59:07.708: I/BookNowLog(23590): UR - Se ha actualizado el usuario correctamente

12-09 17:59:07.718: I/BookNowLog(23590): DB - Se ha abierto la base de datos para escritura

12-09 17:59:07.798: I/BookNowLog(23590): DB - Se ha eliminado el usuario de la base de datos

12-09 17:59:07.853: I/BookNowLog(23590): DB - Se ha abierto la base de datos para escritura

12-09 17:59:07.888: I/BookNowLog(23590): DB - Se ha guardado el usuario de la base de datos

12-09 17:59:11.523: I/BookNowLog(23590): RV - Reservas cargadas correctamente

Acción de cerrar sesión

12-09 18:05:27.158: I/BookNowLog(23590): DB - Se ha abierto la base de datos para escritura

12-09 18:05:27.208: I/BookNowLog(23590): DB - Se ha actualizado el usuario de la base de datos

12-09 18:05:27.228: I/BookNowLog(23590): DB - Se ha abierto la base de datos para lectura

12-09 18:05:27.233: I/BookNowLog(23590): DB - Se ha cargado el usuario de la base de datos

12-09 18:05:27.308: I/BookNowLog(23590): Activity SinginUp iniciada

Acciones de usuario cliente

Acción de registro de usuario

12-09 17:44:30.008: I/BookNowLog(23013): Activity Init iniciada

12-09 17:44:32.233: I/BookNowLog(23013): DB - Se ha abierto la base de datos para lectura

12-09 17:44:32.253: I/BookNowLog(23013): DB - Se ha cargado el usuario de la base de datos

12-09 17:44:32.283: I/BookNowLog(23013): SS - Existe una sesión de usuario

12-09 17:44:32.448: I/BookNowLog(23013): Activity SinginUp iniciada

12-09 17:44:46.083: I/BookNowLog(23013): Activity Register iniciada

12-09 17:45:31.253: I/BookNowLog(23013): RG - Se ha registrado correctamente

12-09 17:45:34.453: I/BookNowLog(23013): SN - Se ha iniciado sesión correctamente

12-09 17:45:34.473: I/BookNowLog(23013): DB - Se ha abierto la base de datos para escritura

12-09 17:45:34.528: I/BookNowLog(23013): DB - Se ha eliminado el usuario de la base de datos

12-09 17:45:34.588: I/BookNowLog(23013): DB - Se ha abierto la base de datos para escritura

12-09 17:45:34.618: I/BookNowLog(23013): DB - Se ha guardado el usuario de la base de datos

12-09 17:45:35.098: I/BookNowLog(23013): Activity Main iniciada

12-09 17:45:39.848: I/BookNowLog(23013): RV - Reservas cargadas correctamente

12-09 17:45:43.538: I/BookNowLog(23013): RD - Se han cargado los recordatorios correctamente

Acción de inicio de sesión

12-09 17:49:43.723: I/BookNowLog(23424): Activity Init iniciada

12-09 17:49:46.018: I/BookNowLog(23424): DB - Se ha abierto la base de datos para lectura

12-09 17:49:46.038: I/BookNowLog(23424): DB - Se ha cargado el usuario de la base de datos

12-09 17:49:46.048: I/BookNowLog(23424): SS - Existe una sesión de usuario

12-09 17:49:46.213: I/BookNowLog(23424): Activity SinginUp iniciada

12-09 17:49:56.948: I/BookNowLog(23424): SN - Se ha iniciado sesión correctamente

12-09 17:49:56.958: I/BookNowLog(23424): DB - Se ha abierto la base de datos para escritura

12-09 17:49:57.018: I/BookNowLog(23424): DB - Se ha eliminado el usuario de la base de datos

12-09 17:49:57.083: I/BookNowLog(23424): DB - Se ha abierto la base de datos para escritura

12-09 17:49:57.113: I/BookNowLog(23424): DB - Se ha guardado el usuario de la base de datos

12-09 17:49:57.618: I/BookNowLog(23424): Activity Main iniciada

12-09 17:50:02.383: I/BookNowLog(23424): RV - Reservas cargadas correctamente

12-09 17:50:05.843: I/BookNowLog(23424): RD - Se han cargado los recordatorios correctamente

Acción de realizar reserva de búsqueda por lista

12-09 18:18:17.283: I/BookNowLog(26058): Activity ListServices iniciada

12-09 18:18:32.433: I/BookNowLog(26058): SV - Servicios cargados correctamente

12-09 18:18:42.608: I/BookNowLog(26058): Activity DetailService iniciada

12-09 18:18:46.008: I/BookNowLog(26058): Activity Calendar iniciada

12-09 18:18:49.623: I/BookNowLog(26058): RV - Reservas cargadas correctamente

12-09 18:18:58.573: I/BookNowLog(26058): RV - Reserva grabada correctamente

12-09 18:19:02.783: I/BookNowLog(26058): RV - Reservas cargadas correctamente

12-09 18:19:05.963: I/BookNowLog(26058): RD - Se han cargado los recordatorios correctamente

Acción de realizar reserva de búsqueda por mapa

12-09 18:19:36.958: I/BookNowLog(26058): Activity ListServices iniciada

12-09 18:19:40.093: I/BookNowLog(26058): SV - Servicios cargados correctamente

12-09 18:19:42.868: I/BookNowLog(26058): Activity Selected iniciada

12-09 18:19:42.878: I/BookNowLog(26058): GPS - Se ha creado el localizador manager correctamente

12-09 18:19:42.888: I/BookNowLog(26058): GPS - Se ha cargado el proveedor correctamente

12-09 18:19:42.888: I/BookNowLog(26058): GPS - Se ha cargado la localización correctamente

12-09 18:19:42.973: I/BookNowLog(26058): GPS - Se ha cargado el proveedor correctamente

12-09 18:19:42.978: I/BookNowLog(26058): GPS - Se ha cargado la localización correctamente

12-09 18:19:46.108: I/BookNowLog(26058): SV - Servicios cargados correctamente

12-09 18:19:51.603: I/BookNowLog(26058): SV - Servicios cargados correctamente

12-09 18:19:59.313: I/BookNowLog(26058): Activity DetailService iniciada

12-09 18:20:00.803: I/BookNowLog(26058): Activity Calendar iniciada

12-09 18:20:03.248: I/BookNowLog(26058): RV - Reservas cargadas correctamente

12-09 18:20:18.423: I/BookNowLog(26058): RV - Reserva grabada correctamente

12-09 18:20:21.588: I/BookNowLog(26058): RV - Reservas cargadas correctamente

12-09 18:20:23.973: I/BookNowLog(26058): RD - Se han cargado los recordatorios correctamente

Acción de consultar reserva y su localización

12-09 18:28:25.588: I/BookNowLog(26855): Activity DetailReserve iniciada

12-09 18:28:30.658: I/BookNowLog(26855): SV - Servicios cargados correctamente

12-09 18:28:31.083: I/BookNowLog(26855): Activity Selected iniciada

12-09 18:28:31.088: I/BookNowLog(26855): GPS - Se ha creado el localizador manager correctamente

12-09 18:28:31.098: I/BookNowLog(26855): GPS - Se ha cargado el proveedor correctamente

12-09 18:28:31.098: I/BookNowLog(26855): GPS - Se ha cargado la localización correctamente

12-09 18:28:31.138: I/BookNowLog(26855): GPS - Se ha cargado el proveedor correctamente

12-09 18:28:31.138: I/BookNowLog(26855): GPS - Se ha cargado la localización correctamente

Acción de eliminar una reserva

12-09 18:32:32.348: I/BookNowLog(26855): Activity DetailReserve iniciada

12-09 18:32:40.108: I/BookNowLog(26855): RV - Reserva eliminada correctamente

12-09 18:32:42.678: I/BookNowLog(26855): RV - Reservas cargadas correctamente

12-09 18:32:45.568: I/BookNowLog(26855): RD - Se han cargado los recordatorios correctamente

Acción de realizar un recordatorio

12-09 18:27:24.408: I/BookNowLog(26855): Activity DetailReserve iniciada

12-09 18:27:33.023: I/BookNowLog(26855): RD - Se ha grabado el recordatorio correctamente

12-09 18:27:37.193: I/BookNowLog(26855): RD - Se han cargado los recordatorios correctamente

Acción de eliminar un recordatorio

12-09 18:29:56.168: I/BookNowLog(26855): RD - Se ha eliminado el recordatorio correctamente

12-09 18:29:59.458: I/BookNowLog(26855): RD - Se han cargado los recordatorios correctamente

12-09 18:30:01.478: I/BookNowLog(26855): RD - Se han cargado los recordatorios correctamente

Acción de aviso de recordatorio y actualización

12-09 18:52:42.205: I/BookNowLog(4883): Activity Init iniciada

12-09 18:52:44.460: I/BookNowLog(4883): DB - Se ha abierto la base de datos para lectura

12-09 18:52:44.460: I/BookNowLog(4883): DB - Se ha cargado el usuario de la base de datos

12-09 18:52:44.470: I/BookNowLog(4883): SS - Existe una sesión de usuario

12-09 18:52:45.005: I/BookNowLog(4883): Activity Main iniciada

12-09 18:52:50.295: I/BookNowLog(4883): RV - Reservas cargadas correctamente

12-09 18:52:52.320: I/BookNowLog(4883): RD - Se han cargado los recordatorios correctamente

12-09 18:53:39.095: I/BookNowLog(4883): RD - Se ha actualizado el recordatorio correctamente

Acción de editar cuenta de usuario

12-09 18:36:42.808: I/BookNowLog(26855): UR - Se ha actualizado el usuario correctamente

12-09 18:36:42.823: I/BookNowLog(26855): DB - Se ha abierto la base de datos para escritura

12-09 18:36:42.903: I/BookNowLog(26855): DB - Se ha eliminado el usuario de la base de datos

12-09 18:36:42.968: I/BookNowLog(26855): DB - Se ha abierto la base de datos para escritura

12-09 18:36:42.993: I/BookNowLog(26855): DB - Se ha guardado el usuario de la base de datos

Acción de cerrar sesión

12-09 18:37:52.053: I/BookNowLog(26855): DB - Se ha abierto la base de datos para escritura

12-09 18:37:52.108: I/BookNowLog(26855): DB - Se ha actualizado el usuario de la base de datos

12-09 18:37:52.143: I/BookNowLog(26855): DB - Se ha abierto la base de datos para lectura

12-09 18:37:52.148: I/BookNowLog(26855): DB - Se ha cargado el usuario de la base de datos

12-09 18:37:52.208: I/BookNowLog(26855): Activity SinginUp iniciada

Resultados de pruebas de usabilidad manuales

ACCESO A APLICACIÓN	
Propósito	Verificar los diferentes accesos a la aplicación

Núm.	Acción a verificar	Resultado esperado	Verificación
001	Alta de usuario con datos incorrectos.	No nos deja realizar el registro y nos muestra un aviso.	OK
002	Alta de usuario el mail ya existe en la base de datos.	No nos deja realizar el registro y nos muestra un aviso.	OK
003	Inicio de sesión con cliente usuario	La aplicación se abre en modo cliente	OK
004	Inicio de sesión en modo empresa	La aplicación de abre en modo empresa	OK
005	Inicio de sesión con datos incorrectos	No nos deja iniciar sesión y nos da un aviso.	OK
006	Se inicia la aplicación con problemas de acceso a internet.	Se muestra un cuadro de diálogo informando y se muestra información en las diferentes pantallas que no	OK

		pueden mostrar datos y se bloquean controles.	
007	Tenemos una notificación de recordatorio con la aplicación cerrada y la pulsamos.	Se abre la aplicación y nos muestra la notificación.	OK

APLICACIÓN EN MODO EMPRESA	
Propósito	Verificar las diferentes acciones que se realizan en la aplicación en modo empresa.

Núm.	Acción a verificar	Resultado esperado	Verificación
Apartado Agenda			
001	Accedemos al apartado de agenda y no tenemos ninguna reserva para ese día.	No nos muestra ninguna reserva y vemos un texto informativo.	OK
002	Accedemos al apartado de agenda y tenemos reservas para ese día.	Nos muestra la reserva o lista de reservas.	OK
003	En la pantalla de agenda pulsamos el icono de agenda.	Se muestra el calendario en un cuadro de diálogo, si tenemos reservas salen indicadas con un icono.	OK
004	En el cuadro de diálogo de calendario pulsamos sobre un día.	El cuadro de diálogo se cierra y en la pantalla de agenda se muestran las reservas para ese día.	OK
005	En el cuadro de diálogo de calendario pulsamos los iconos de recorrido.	Se actualiza la información del calendario con los datos del mes correspondiente.	
Apartado Reservar			
006	Accedemos al apartado de realizar reservas.	Se muestra el calendario, si tenemos reservas salen indicadas con un icono.	OK
007	Hay un día completo en el calendario.	Sale indicado en el calendario con un icono.	OK
008	Pulsamos un día que está completo.	No nos deja realizar la reserva y nos informa.	OK

009	Hay horas completas para el mismo día y la misma hora y el número de empleados ya cubre todas las reservas que se pueden realizar en esa hora.	En el cuadro de diálogo salen las horas deshabilitadas.	OK
010	Pulsamos en el diálogo de selección de hora para el mismo día en el que nos encontramos una hora igual o inferior a la actual.	No nos deja realizar la reserva y nos informa.	OK
011	Pulsamos una hora sobre el cuadro de diálogo y pulsamos aceptar.	Se muestra un cuadro de diálogo para introducir el nombre de la reserva.	OK
012	En el cuadro de introducir nombre pulsamos aceptar.	Se realiza la reserva y nos informa con un cuadro de diálogo con información de la reserva.	OK
013	En la pantalla de calendario pulsamos los iconos de recorrido.	Se actualiza la información del calendario con los datos del mes correspondiente.	OK
Apartado cuenta de usuario			
014	Sobre la pantalla de cuenta de usuario pulsamos editar.	Se habilitan los controles edittext y podemos editar los datos, aparecen los controles aceptar y cancelar.	OK
015	Pulsamos aceptar y ese usuario no tiene reservas.	Se actualizan los datos de usuario.	OK
016	Pulsamos el botón de cerrar sesión.	Se cierra la sesión y vamos a la pantalla inicial para iniciar una nueva sesión.	OK

APLICACIÓN EN MODO CLIENTE	
Propósito	Verificar las diferentes acciones que se realizan en la aplicación en modo cliente.

Núm.	Acción a verificar	Resultado esperado	Verificación
Apartado Búsqueda			
001	Seleccionamos un tipo de servicio	No carga ningún servicio y nos informa en un cuadro	OK

	que no tiene servicios asociados.	de texto.	
002	No tenemos ningún servicio relacionado y le pulsamos la opción de búsqueda por mapa.	Abre el mapa pero no nos muestra ningún servicio.	OK
003	Seleccionamos un tipo de servicio que tiene servicios asociados.	Muestra los servicios en pantalla.	OK
004	Tenemos servicios relacionados y pulsamos la búsqueda por mapa con el GPS activado.	Muestra el mapa centrado en la posición del dispositivo y muestra los servicios que hay 20km a la redonda.	OK
005	Tenemos servicios relacionados y pulsamos la búsqueda por mapa con el GPS desactivado.	Muestra el mapa centrado en un punto por defecto y muestra todos los servicios.	OK
006	Realizamos una búsqueda por mapa con el GPS desactivado la primera vez.	Muestra el mapa con un diálogo que nos pregunta si queremos activar el GPS.	OK
007	Aceptamos el cuadro de diálogo que nos pregunta si queremos activar el GPS.	Se abre la pantalla de configuración del móvil para activar el GPS.	OK
008	En el mapa pulsamos sobre un punto de servicio.	Se abre un diálogo preguntándonos si queremos ver detalles del servicio.	OK
009	Pulsamos aceptar en el cuadro de diálogo que nos pregunta si queremos ver los detalles del servicio.	Nos muestra los detalles del servicio.	OK
010	Desde la lista de servicios pulsamos un servicio.	Nos muestra los detalles del servicio.	OK
011	En la pantalla de lista de servicios introducimos un nombre de servicio existente y pulsamos el botón de búsqueda.	Nos muestra los servicios que contengan el texto.	OK
012	Desde la pantalla de detalles de servicio pulsamos el botón reservar.	Se abre el calendario para con la fecha actual si tenemos reservas salen indicadas en el calendario.	OK
013	Pulsamos un día que es domingo para realizar la reserva.	No nos deja realizar la reserva y nos informa que el servicio no abre ese día.	OK

014	Pulsamos un día que es sábado para realizar la reserva y ese servicio abre 5 días.	No nos deja realizar la reserva y nos informa que el servicio no abre ese día.	OK
015	Pulsamos un día que es sábado para realizar la reserva y ese servicio abre 6 días.	Si el día no está completo y no tenemos ninguna reserva nos muestra el cuadro de diálogo para elegir la hora de la reserva.	OK
016	Tenemos una reserva en ese mes.	Se dibuja en el calendario el día de la reserva con un icono.	OK
017	Pulsamos un día que tenemos ya una reserva.	No nos deja realizar la reserva y nos informa.	OK
018	Hay un día completo.	Sale indicado en el calendario con un icono.	OK
019	Pulsamos un día que está completo.	No nos deja realizar la reserva y nos informa.	OK
020	Hay horas completas para el mismo día y la misma hora y el número de empleados ya cubre todas las reservas que se pueden realizar en esa hora.	En el cuadro de diálogo salen las horas deshabilitadas.	OK
021	Pulsamos en el diálogo de selección de hora para el mismo día en el que nos encontramos una hora igual o inferior a la actual.	No nos deja realizar la reserva y nos informa.	OK
022	Pulsamos una hora sobre el cuadro de diálogo y pulsamos aceptar.	Se realiza la reserva y nos informa con un cuadro de diálogo con información de la reserva.	OK
023	Pulsamos aceptar sobre el cuadro de diálogo de información de la reserva.	Volvemos a la pantalla inicial.	OK
024	Pulsamos una hora sobre el cuadro de diálogo y pulsamos aceptar.	Se realiza la reserva y nos informa con un cuadro de diálogo con información de la reserva.	OK
025	En la pantalla de calendario pulsamos los iconos de recorrido.	Se actualiza la información del calendario con los datos del mes correspondiente.	OK
Apartado Mis reservas			

026	En la pantalla principal seleccionamos el icono para consultar mis reservas y no tenemos reservas.	No se muestran reservas y se muestra un texto informativo.	OK
027	En la pantalla principal seleccionamos el icono para consultar mis reservas y tenemos reservas.	Se muestra la lista de reservas del usuario.	OK
028	Pulsamos sobre una reserva en la lista de reservas.	Se muestra la información detallada de la reserva	OK
029	En la pantalla de detalle de reserva pulsamos el botón mapa.	Se muestra el mapa con la localización del servicio.	OK
030	En la pantalla de detalle de reserva pulsamos el botón recordatorio.	Se abre un cuadro de diálogo para realizar en recordatorio.	OK
031	En el cuadro de diálogo para realizar un recordatorio pulsamos aceptar.	Se realiza el recordatorio y nos informa.	OK
032	En el cuadro de diálogo para realizar un recordatorio pulsamos aceptar para realizar sobre una reserva que ya tenemos recordatorio.	No nos deja hacer el recordatorio y nos informa que ya tenemos un recordatorio para esa reserva.	OK
033	En la pantalla de detalles de reserva pulsamos el botón eliminar.	Se muestra un cuadro de diálogo que nos pregunta si queremos eliminar la reserva.	OK
034	Sobre el cuadro de diálogo que nos pregunta si queremos eliminar la reserva pulsamos aceptar.	La reserva se elimina y nos informa.	OK
035	En la pantalla de lista de reservas pulsamos el icono de recordatorios y no tenemos recordatorios.	No se muestra ningún recordatorio.	OK
036	En la pantalla de lista de reservas pulsamos el icono de recordatorios y tenemos recordatorios.	Se muestra un cuadro de diálogo con una lista de recordatorios.	OK
037	Sobre el cuadro de diálogo de recordatorio realizamos una pulsación larga sobre un recordatorio.	Se muestra un cuadro de diálogo preguntándonos si queremos eliminar el recordatorio.	OK
038	Sobre el cuadro de diálogo de eliminar recordatorio pulsamos aceptar.	Se elimina el recordatorio y nos informa.	OK

Apartado cuenta de usuario			
039	Sobre la pantalla de cuenta de usuario pulsamos editar.	Se habilitan los controles edittext y podemos editar los datos, aparecen los controles aceptar y cancelar.	OK
040	Pulsamos aceptar y ese usuario tiene reservas.	No nos deja editar y nos informa.	OK
041	Pulsamos aceptar y ese usuario no tiene reservas.	Se actualizan los datos de usuario.	OK
042	Pulsamos el botón de cerrar sesión.	Se cierra la sesión y vamos a la pantalla inicial para iniciar una nueva sesión.	OK

Resultados de pruebas de test unitarias

12-09 21:23:42.454: I/BookNowLog(17759): Activity Init iniciada

12-09 21:23:43.029: I/BookNowLog(17759): TEST - Iniciamos test del modulo Tools

12-09 21:23:43.029: I/BookNowLog(17759): TEST - Comprobación de función getStringTime

12-09 21:23:43.034: I/BookNowLog(17759): TEST - La función getStringTime ha finalzado con éxito

12-09 21:23:43.034: I/BookNowLog(17759): TEST - Comprobación de función compareTime

12-09 21:23:43.034: I/BookNowLog(17759): TEST - La función compareTime ha finalizado con éxito

12-09 21:23:43.034: I/BookNowLog(17759): TEST - Comprobación de función compareData

12-09 21:23:43.059: I/BookNowLog(17759): TEST - La función compareData para fechas iguales ha finalizado con éxito

12-09 21:23:43.084: I/BookNowLog(17759): TEST - La función compareData para fechas inferiores ha finalizado con éxito

12-09 21:23:43.089: I/BookNowLog(17759): TEST - La función compareData para fechas superiores ha finalizado con éxito

12-09 21:23:43.089: I/BookNowLog(17759): TEST - Comprobación de función CalculateTotalDays

12-09 21:23:43.124: I/BookNowLog(17759): TEST - La función CalculateTotalDays ha finalizado con éxito

12-09 21:23:43.124: I/BookNowLog(17759): TEST - Comprobación de función CheckUserData

12-09 21:23:43.124: I/BookNowLog(17759): TEST - La función CheckUserData para campos nulos ha finalizado con éxito

12-09 21:23:43.124: I/BookNowLog(17759): TEST - La función CheckUserData que verifica la contraseña ha finalizado con éxito

12-09 21:23:43.124: I/BookNowLog(17759): TEST - La función CheckUserData que verifica el mail ha finalizado con éxito

12-09 21:23:43.124: I/BookNowLog(17759): TEST - La función CheckUserData que verifica los datos ha finalizado con éxito

12-09 21:23:43.124: I/BookNowLog(17759): TEST - Iniciamos el test del módulo activityMessage

12-09 21:23:43.124: I/BookNowLog(17759): TEST - Asignamos el mensaje

12-09 21:23:43.124: I/BookNowLog(17759): TEST - Enviamos el mensaje

12-09 21:23:43.124: I/BookNowLog(17759): TEST - Si la operación termina recibiremos una respuesta en las siguientes líneas

12-09 21:23:56.994: I/BookNowLog(17759): TEST - El módulo activityMessage ha enviado el mensaje con éxito

12-09 21:23:43.124: I/BookNowLog(17759): TEST - Iniciamos el test del módulo Geo

12-09 21:23:43.144: I/BookNowLog(17759): GPS - Se ha creado el localizador manager correctamente

12-09 21:23:43.144: I/BookNowLog(17759): TEST - El objeto se ha creado correctamente

12-09 21:23:43.144: I/BookNowLog(17759): TEST - Comprobación de la función Init

12-09 21:23:43.144: I/BookNowLog(17759): GPS - Se ha creado el localizador manager correctamente

12-09 21:23:43.144: I/BookNowLog(17759): TEST - Comprobación de la función GetLocation

12-09 21:23:43.194: I/BookNowLog(17759): GPS - Se ha cargado el proveedor correctamente

12-09 21:23:43.194: I/BookNowLog(17759): GPS - Se ha cargado la localización correctamente

12-09 21:23:43.194: I/BookNowLog(17759): TEST - La función GetLocation ha finalizado con éxito

12-09 21:23:43.194: I/BookNowLog(17759): TEST - Comprobación de la función GetGpsEnabled con GPS activado

12-09 21:23:43.194: I/BookNowLog(17759): TEST - La función GetGpsEnabled ha finalizado con éxito.

12-09 21:23:43.194: I/BookNowLog(17759): TEST - Iniciamos el test del módulo ItemServices

12-09 21:23:43.194: I/BookNowLog(17759): TEST - La primera prueba ha finalizado con éxito

12-09 21:23:43.194: I/BookNowLog(17759): TEST - La segunda prueba ha finalizado con éxito

12-09 21:23:43.194: I/BookNowLog(17759): TEST - Iniciamos el test del módulo MyCalendarAdapter

12-09 21:23:43.194: I/BookNowLog(17759): TEST - Comprobación de la función returnDay

12-09 21:23:43.194: I/BookNowLog(17759): TEST - La función returnDay ha finalizado con éxito

12-09 21:23:43.194: I/BookNowLog(17759): TEST - Iniciamos el test del módulo MyPoint

12-09 21:23:43.194: I/BookNowLog(17759): TEST - Asignamos un nuevo GeoPoint y OverlayItem

12-09 21:23:43.234: I/BookNowLog(17759): TEST - La asignación del GeoPoint ha finalizado con éxito

12-09 21:23:43.234: I/BookNowLog(17759): TEST - La asignación del OverlayItem ha finalizado con éxito

12-09 21:23:43.234: I/BookNowLog(17759): TEST - Comprobación de la función erasePoints

12-09 21:23:43.234: I/BookNowLog(17759): TEST - La función erasePoints ha finalizado con éxito

12-09 21:23:56.999: I/BookNowLog(17759): Activity Register iniciada

12-09 21:23:57.144: I/BookNowLog(17759): TEST - Iniciamos el test módulo Register

12-09 21:23:57.144: I/BookNowLog(17759): TEST - Carga de controles

12-09 21:23:57.144: I/BookNowLog(17759): TEST - Controles cargados correctamente

12-09 21:23:57.144: I/BookNowLog(17759): TEST - Test del módulo Register finalizado

12-09 21:23:57.309: I/BookNowLog(17759): Activity ListServices iniciada

12-09 21:23:57.309: I/BookNowLog(17759): TEST - Iniciamos el test del módulo ListServicesActivity

12-09 21:23:57.309: I/BookNowLog(17759): TEST - Carga de controles

12-09 21:23:57.309: I/BookNowLog(17759): TEST - Controles cargados correctamente

12-09 21:23:57.309: I/BookNowLog(17759): TEST - Test del módulo ListServicesActivity finalizado

12-09 21:23:57.439: I/BookNowLog(17759): Activity DetailService iniciada

12-09 21:23:57.439: I/BookNowLog(17759): TEST - Iniciamos el test del módulo DetailService

12-09 21:23:57.439: I/BookNowLog(17759): TEST - Carga de controles

12-09 21:23:57.439: I/BookNowLog(17759): TEST - Controles cargados correctamente

12-09 21:23:57.439: I/BookNowLog(17759): TEST - Test del módulo DetailService finalizado

12-09 21:23:57.589: I/BookNowLog(17759): Activity DetailReserve iniciada

12-09 21:23:57.589: I/BookNowLog(17759): TEST - Iniciamos el test módulo DetailReserve

12-09 21:23:57.589: I/BookNowLog(17759): TEST - Carga de controles

12-09 21:23:57.594: I/BookNowLog(17759): TEST - Controles cargados correctamente

12-09 21:23:57.594: I/BookNowLog(17759): TEST - Test del módulo DetailReserve finalizado

12-09 21:23:57.939: I/BookNowLog(17759): Activity CompanyMain iniciada

12-09 21:23:57.939: I/BookNowLog(17759): TEST - Iniciamos el test del módulo CompanyMainActivity

12-09 21:23:57.939: I/BookNowLog(17759): TEST - Carga de controles

12-09 21:23:57.944: I/BookNowLog(17759): TEST - Controles cargados correctamente

12-09 21:23:57.944: I/BookNowLog(17759): TEST - Deshabilitar controles

12-09 21:23:57.949: I/BookNowLog(17759): TEST - Controles deshabilitados correctamente

12-09 21:23:57.949: I/BookNowLog(17759): TEST - Test del módulo CompanyMainActivity finalizado

12-09 21:23:58.069: I/BookNowLog(17759): Activity Calendar iniciada

12-09 21:23:58.074: I/BookNowLog(17759): TEST - Iniciamos el test del módulo CalendarActivity

12-09 21:23:58.074: I/BookNowLog(17759): TEST - Carga de controles

12-09 21:23:58.074: I/BookNowLog(17759): TEST - Controles cargados correctamente

12-09 21:23:58.074: I/BookNowLog(17759): TEST - Test del módulo CalendarActivity finalizado

12-09 21:23:58.534: I/BookNowLog(17759): Activity Main iniciada

12-09 21:23:58.539: I/BookNowLog(17759): TEST - Iniciamos el test del módulo MainActivity

12-09 21:23:58.539: I/BookNowLog(17759): TEST - Carga de controles

12-09 21:23:58.544: I/BookNowLog(17759): TEST - Controles cargados correctamente

12-09 21:23:58.544: I/BookNowLog(17759): TEST - Comprobamos la carga de ítems en la función ShowService

12-09 21:23:58.549: I/BookNowLog(17759): TEST - Ítems cargados correctamente

12-09 21:23:58.549: I/BookNowLog(17759): TEST - Deshabilitar controles

12-09 21:23:58.554: I/BookNowLog(17759): TEST - Controles deshabilitados correctamente

12-09 21:23:58.554: I/BookNowLog(17759): TEST - Test del módulo MainActivity finalizado

Apéndice D. Manual de uso.

En este apendice se va a explicar cómo instalar la aplicación y se detalla un manual de uso de la misma.

El proyecto está compuesto por dos archivos binarios de instalación *.apk estos archivos los encontramos dentro del directorio \bin de cada aplicación.

bin \ BookNow.apk : Archivo binario de la aplicación principal.

bin \ ManagementBM.apk : Aplicación de gestión para introducir nuevos usuarios del tipo empresa.

Para instalar las aplicaciones tenemos que grabar los archivos en nuestro dispositivo en algún directorio donde después podamos localizarlos.

Podemos cargarlos mediante USB o en la tarjeta de memoria externa del dispositivo.

Una vez tengamos los archivos cargados en el dispositivo abriremos el directorio y los ejecutaremos para proceder a la instalación.

Veremos una pantalla como la siguiente:

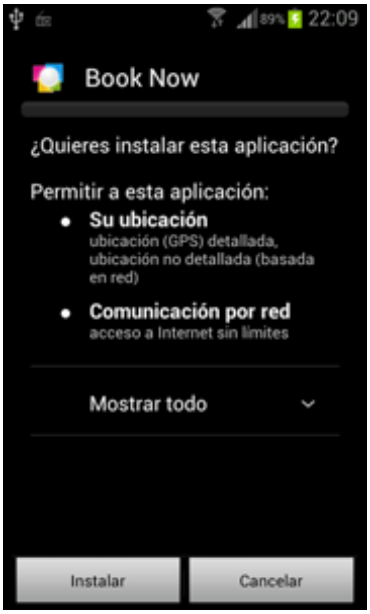


Ilustración 37. Pantalla de instalación

Procedemos a la instalación si las aplicaciones se instalan correctamente las encontraremos como acceso directo en las ventanas de navegación de nuestros dispositivos con los nombres BookNow y ManagementBM los iconos que las identifican son los siguientes:

Ilustración 38. Iconos de aplicación

Para empezar a probar la aplicación hay creados varios usuarios por defecto o se pueden registrar nuevos usuarios.

Usuarios por defecto:

Cliente:

E-mail: client@default.com

Contraseña: 1234

Empresa:

E-mail: company@default.com

Contraseña: 1234

Manual de uso de aplicación ManagementBM

Esta aplicación la utilizaremos para registrar nuevos usuarios de tipo empresa con los correspondientes datos de su negocio.

La aplicación solo consta de una pantalla donde tendremos que introducir los datos del usuario.

En los datos tenemos que prestar principal importancia en el campo número de empleados donde tendremos que poner los empleados que pueden atender a clientes en cada hora, por ejemplo si tenemos 2 usuarios para esa empresa se podrán realizar 2 reservas a la misma hora.

Y también hay que tener especial atención al campo de latitud y longitud donde tenemos que ingresar la localización geográfica de nuestra empresa, los datos de latitud y longitud los podemos conseguir con ayuga de google maps como vemos en la imagen siguiente:

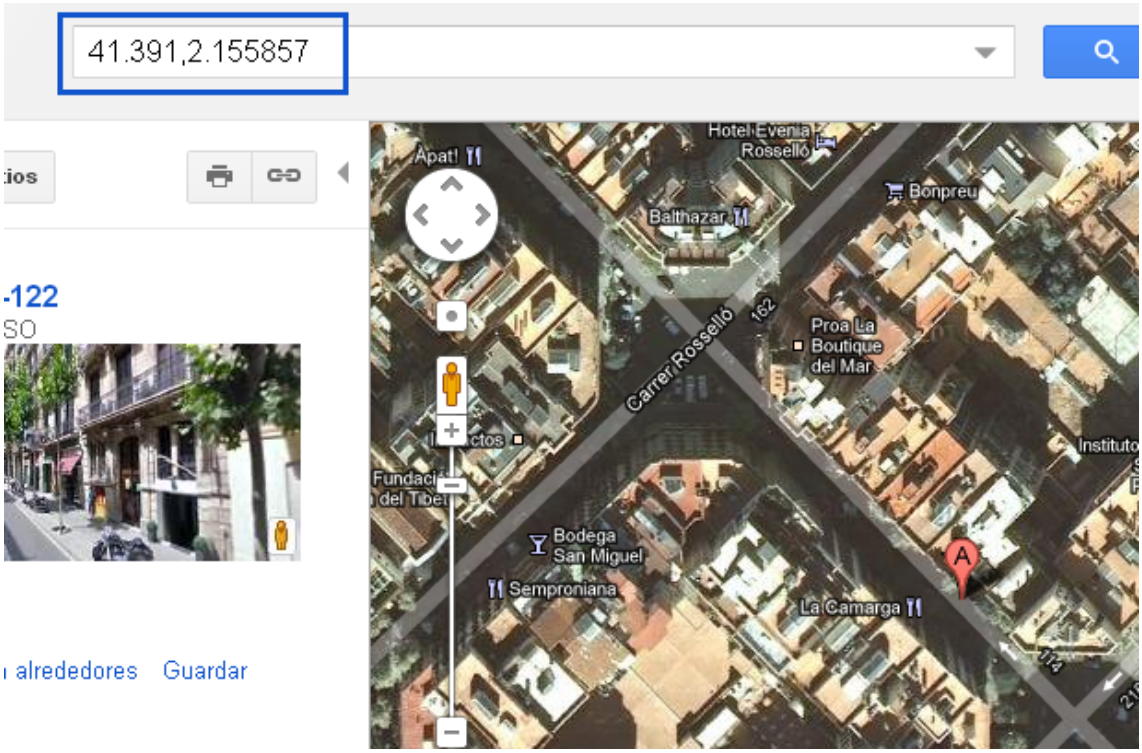


Ilustración 39. Google maps, gestión de coordenadas

Podemos ver una captura de la aplicación managementBM donde tendremos que introducir los datos:

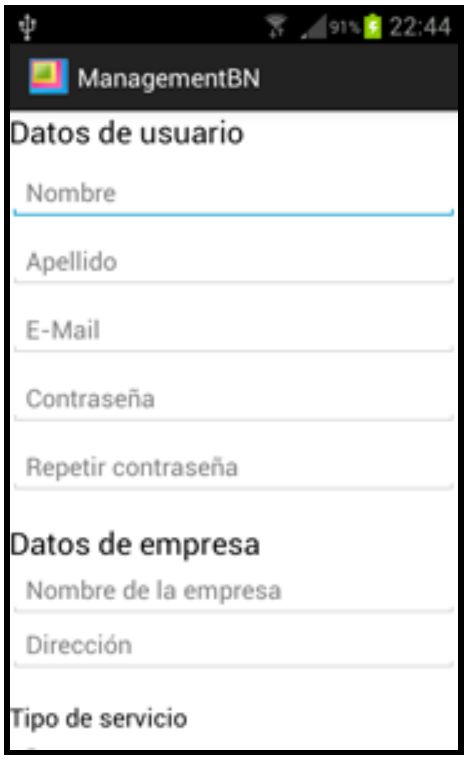


Ilustración 40. Pantalla de ManagementBN

Manual de uso de aplicación BookNow

La primera vez que abrimos la aplicación veremos la pantalla de inicio de sesión, en esta pantalla podemos iniciar la sesión o ir a la pantalla de registro para registrarnos como usuario, desde esta aplicación solo nos podemos registrar como usuario cliente y una vez nos registremos entraremos en la aplicación directamente.

Una vez con la sesión iniciada siempre entraremos automáticamente en la aplicación con el último usuario que ha iniciado la sesión hasta que no la cerremos manualmente.

Ilustración 41. Pantallas de inicio de sesión y registro

Podemos iniciar la aplicación como usuario cliente o como usuario empresa según el tipo de usuario con el que entremos se mostrará la interface correspondiente.

Usuario de tipo empresa

Cuando entremos con el usuario de tipo empresa veremos la siguiente interface gráfica que empezará mostrándonos el apartado de agenda, si tenemos reservas nos mostrará la lista de reservas y si no nos saldrá un texto informativo.

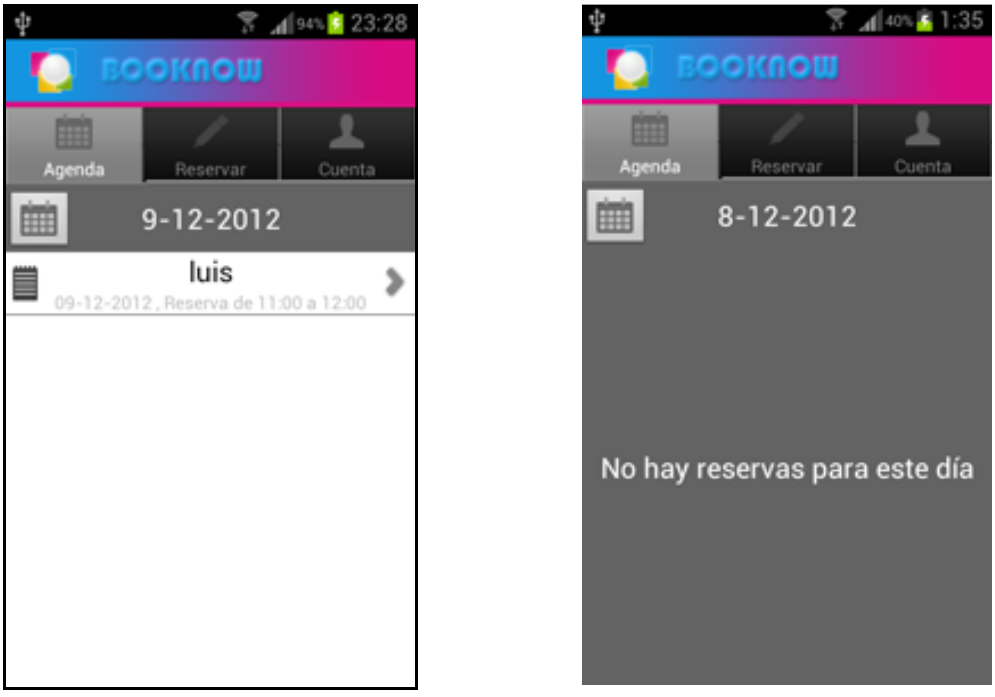


Ilustración 42. Pantalla de agenda, empresa

En esta pantalla podemos pulsar sobre el icono de agenda para ver el calendario y seleccionar el día que queremos consultar, si tenemos reservas se mostrarán con un icono, el día actual se mostrará en azul y si hay algún día completo de mostrará con una franja roja.

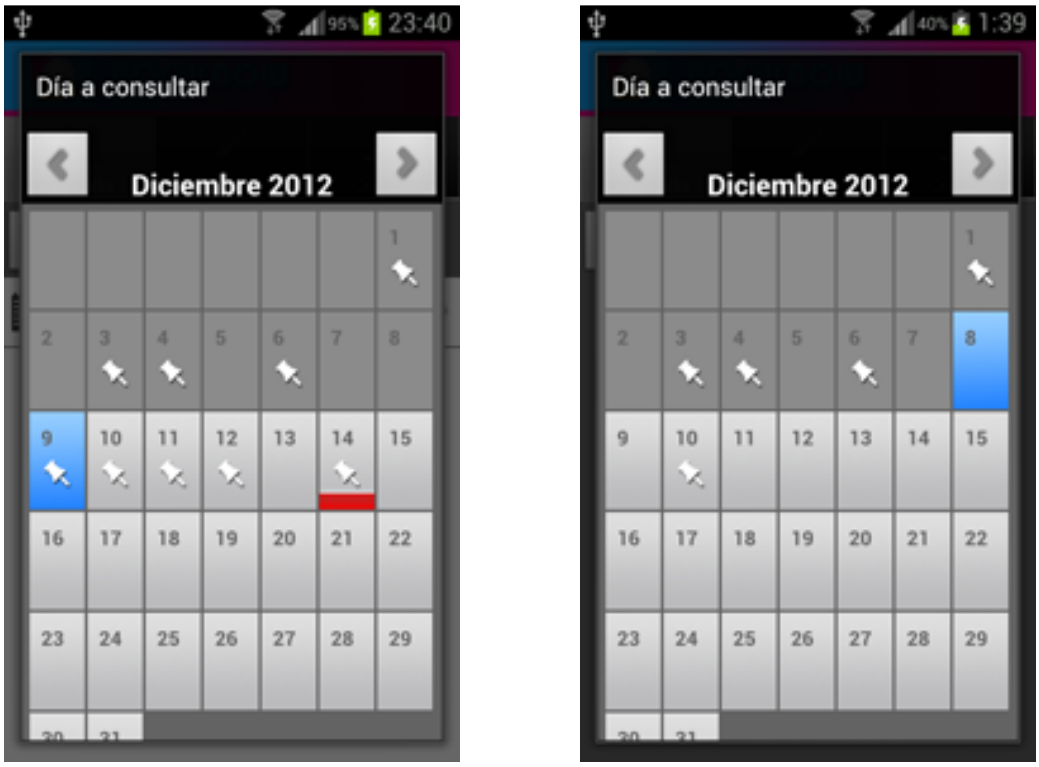


Ilustración 43. Pantallas de calendario, empresa

Leyenda de calendario			
Día del mes	**Día actual**	**Día completo**	**Día con reservas**
			

Ilustración 44. Leyenda de calendario

Si pulsamos sobre una reserva podemos ver la información detallada de la reserva y el código de reserva que enseñará el usuario cuando venga a realizar su servicio.

Desde esta pantalla podemos eliminar la reserva desde el botón eliminar, la aplicación nos pedirá confirmación para eliminar la reserva.

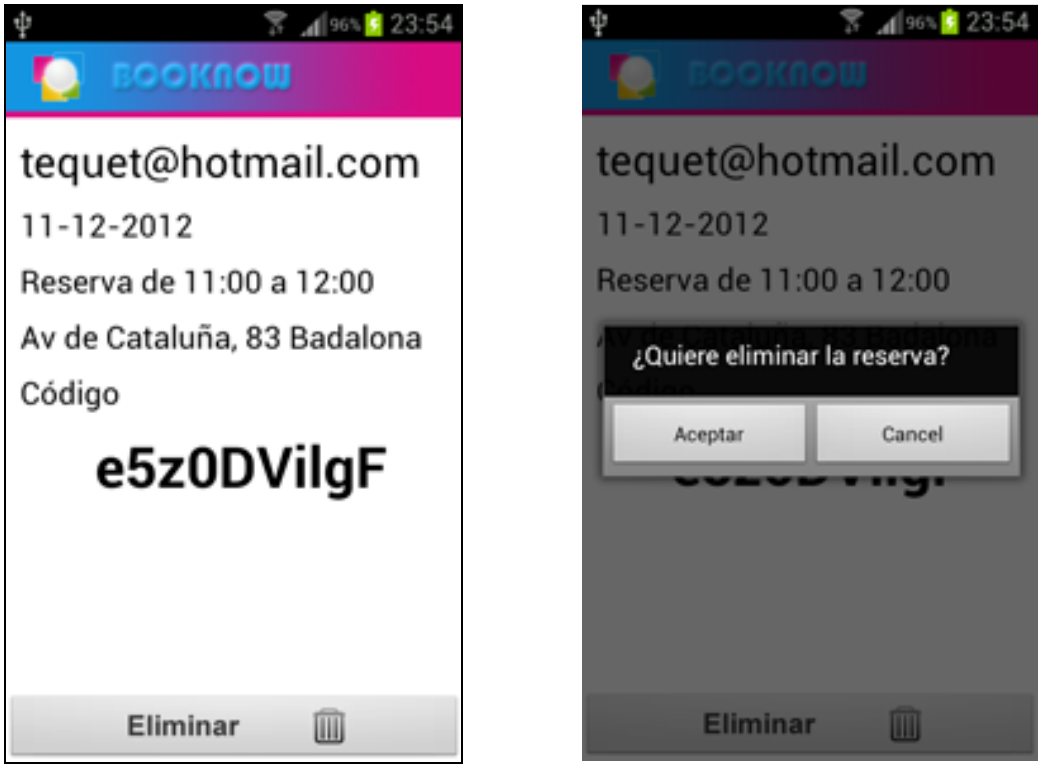


Ilustración 45. Pantallas de detalle de reserva, empresa

Desde el apartado reservar podemos realizar reservas para nuestros clientes editando en nombre de la reserva.

Podremos realizar reservas en días y horas superior a la actual, en días que no estén completos y en horas que no estén completas.

Los usuarios empresa no tienen restricción en reservar horas en días donde supuestamente la empresa está cerrada como podrían ser sábados o domingos.

Cuando pulsemos sobre el día para realizar la reserva se mostrará el cuadro de diálogo para seleccionar la hora.

Ilustración 46. Pantallas de reserva, empresa

Un vez realicemos la reserva se nos mostrará una ventana de información con los datos de la reserva que acabamos de realizar.

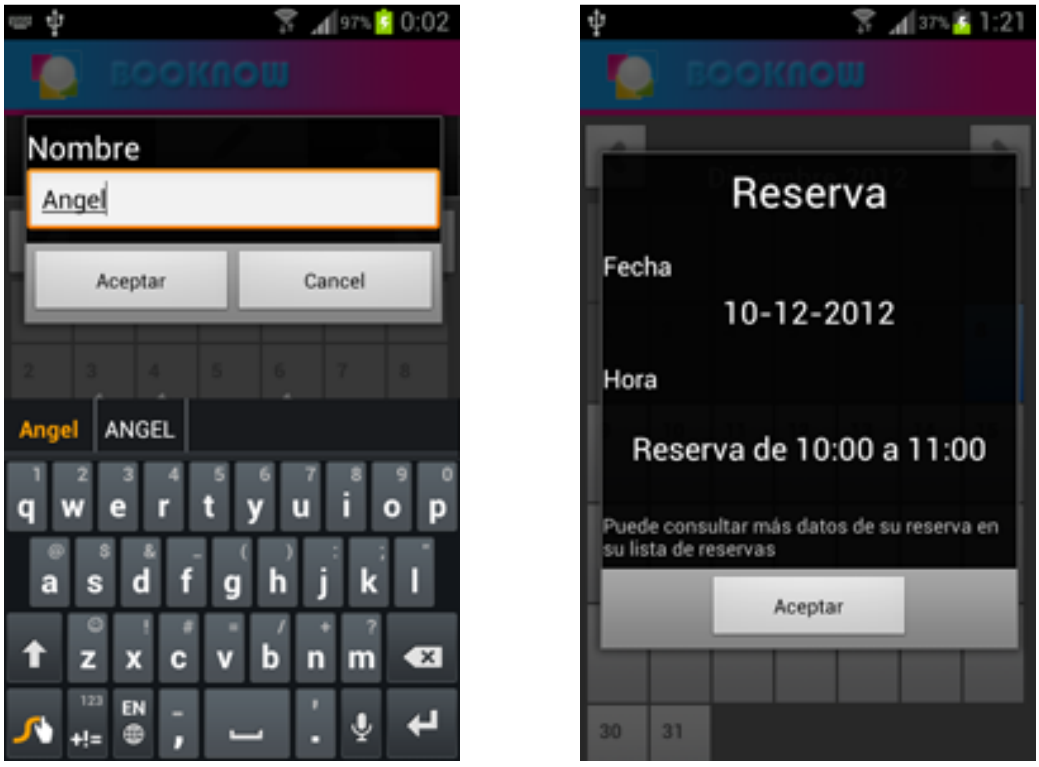


Ilustración 47. Pantallas de realización de reserva, empresa

En el apartado de datos de cuenta de usuario podemos ver los datos de la cuenta, editar los datos o cerrar la sesión.

Ilustración 48. Pantalla de cuenta de usuario

Usuario de tipo cliente

Cuando entramos en la aplicación como usuario cliente empezaremos en el apartado de búsqueda donde podremos ver los tipos de servicios disponibles, si pulsamos sobre un servicio se mostrará una pantalla con la lista de servicios de ese tipo, cada color corresponde a un tipo de servicio y así se mostraran tanto en el menú como en el mapa.

	Servicios jurídicos
	Estética y spa
	Salud
	Servicios de reparación

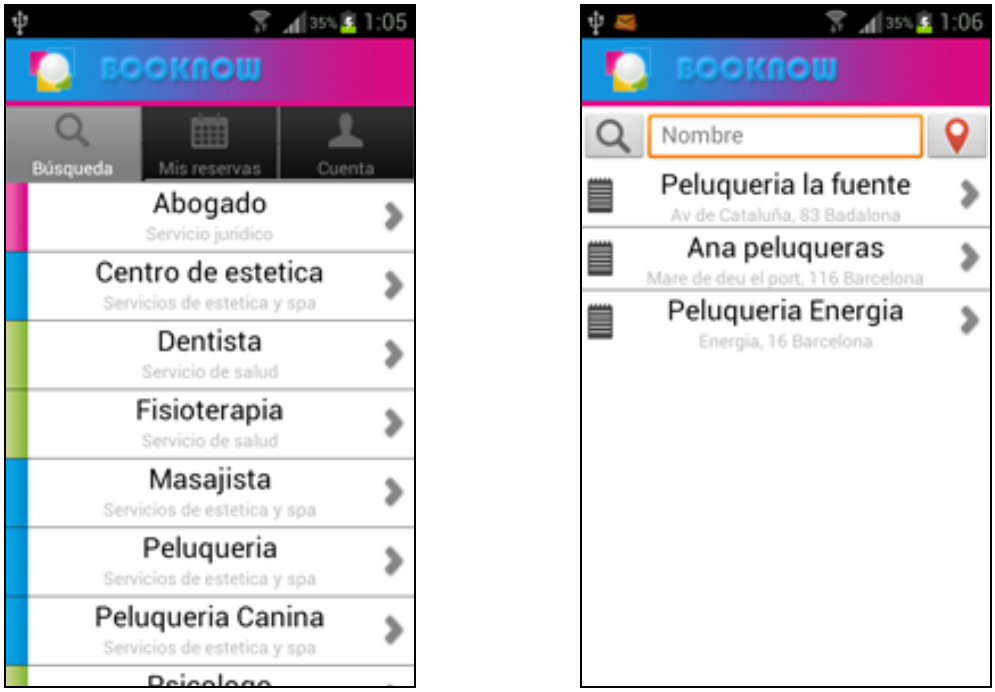


Ilustración 49. Pantallas de servicios, cliente

En la pantalla donde se listan los servicios podemos filtrar la lista por nombre o buscar servicios por localización geográfica.

Cuando entremos en la pantalla de localización geográfica si el GPS esta desactivado se nos mostrará un mensaje para preguntarnos si queremos activar el GPS en caso de aceptar nos llevará a la pantalla de ajustes del dispositivo para poder iniciar el GPS.

Si abrimos el mapa con el GPS activado este se centrará en las coordenadas donde este situado el dispositivo y se mostrarán los servicios del tipo seleccionado que estén en 20km a la redonda.

Si entramos con el GPS desactivado no se mostrará la posición del usuario y el mapa se centrará en un punto por defecto, se mostrarán todos los servicios relacionados con el tipo de servicio seleccionado.

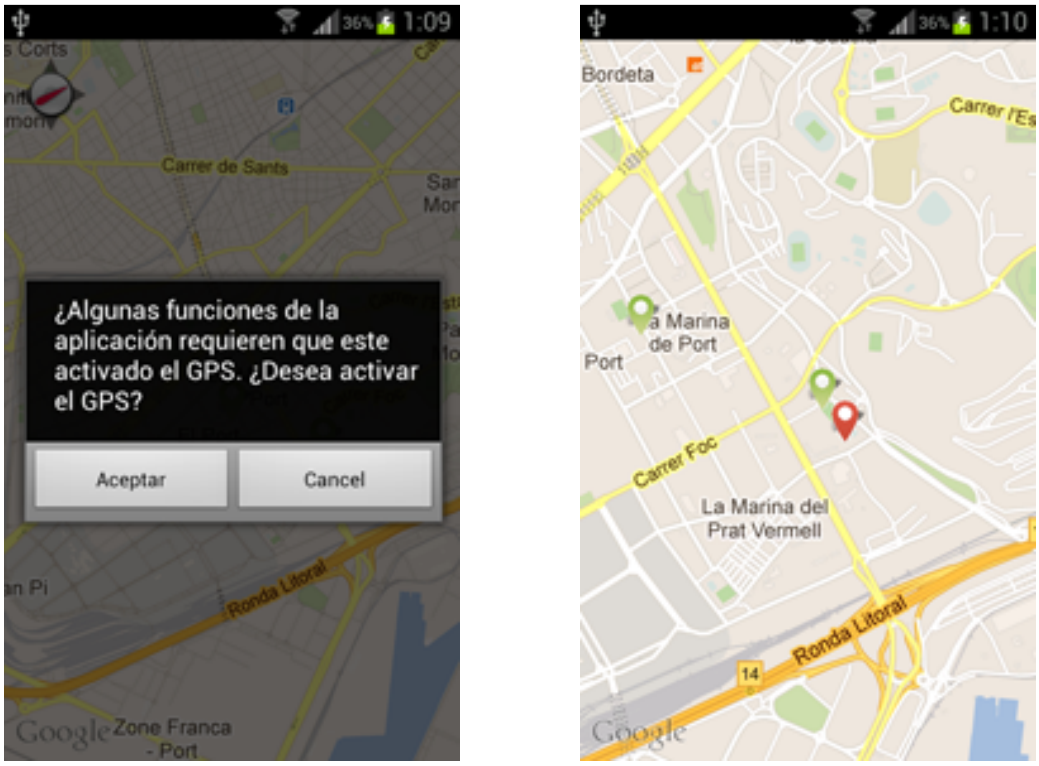


Ilustración 50. Pantallas de localización, cliente

Si pulsamos en el mapa sobre un servicio se abrirá un cuadro de diálogo que nos preguntará si queremos ver los detalles del servicio, también podemos acceder a los detalles del servicio desde la pantalla donde se listan los servicios pulsando sobre el servicio que queramos consultar.

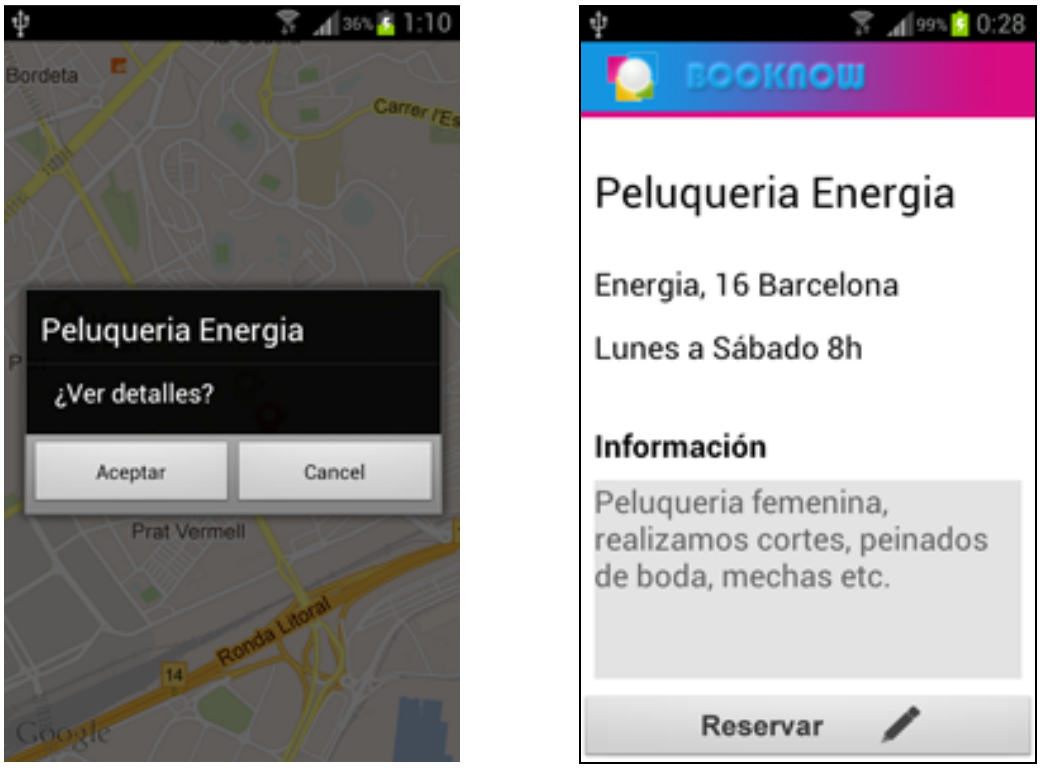


Ilustración 51. Pantallas de detalle de servicio, cliente

Desde la pantalla de detalle de servicio podemos realizar una reserva para ese servicio pulsando el botón reservar se nos abrirá el calendario y el proceso de reserva será el mismo que en el caso de empresa con la diferencia de que solo podemos realizar una reserva por día y no podemos realizar reservas en días que la empresa esté cerrada.

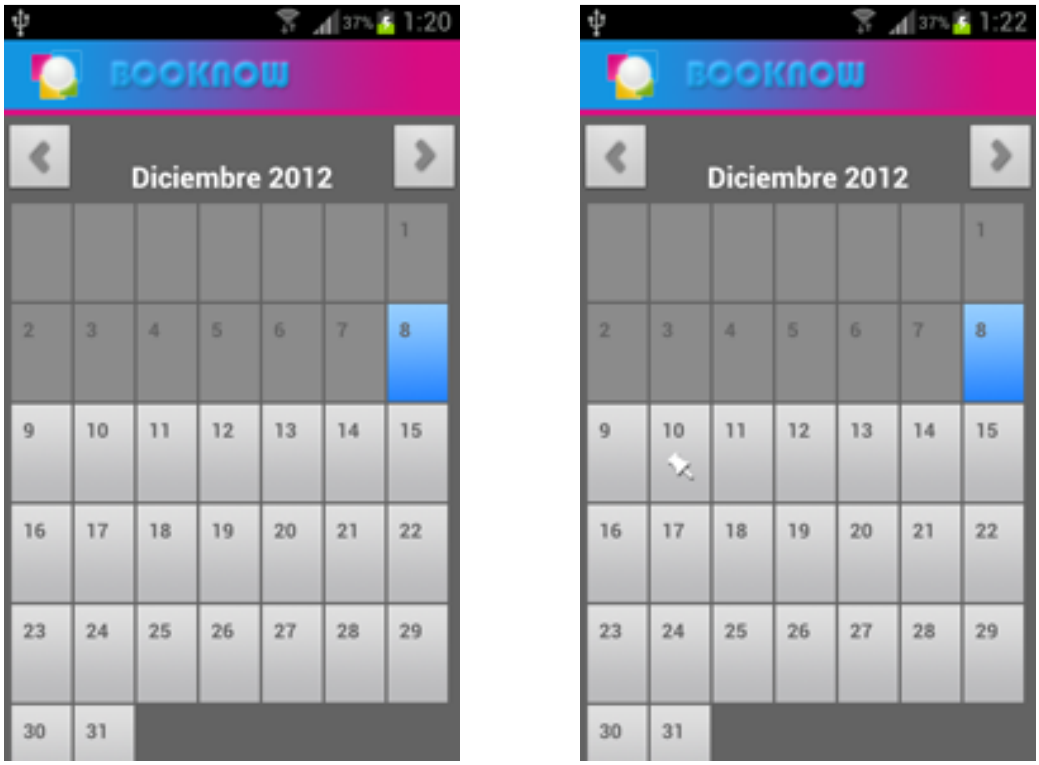


Ilustración 52. Pantallas de calendario, cliente

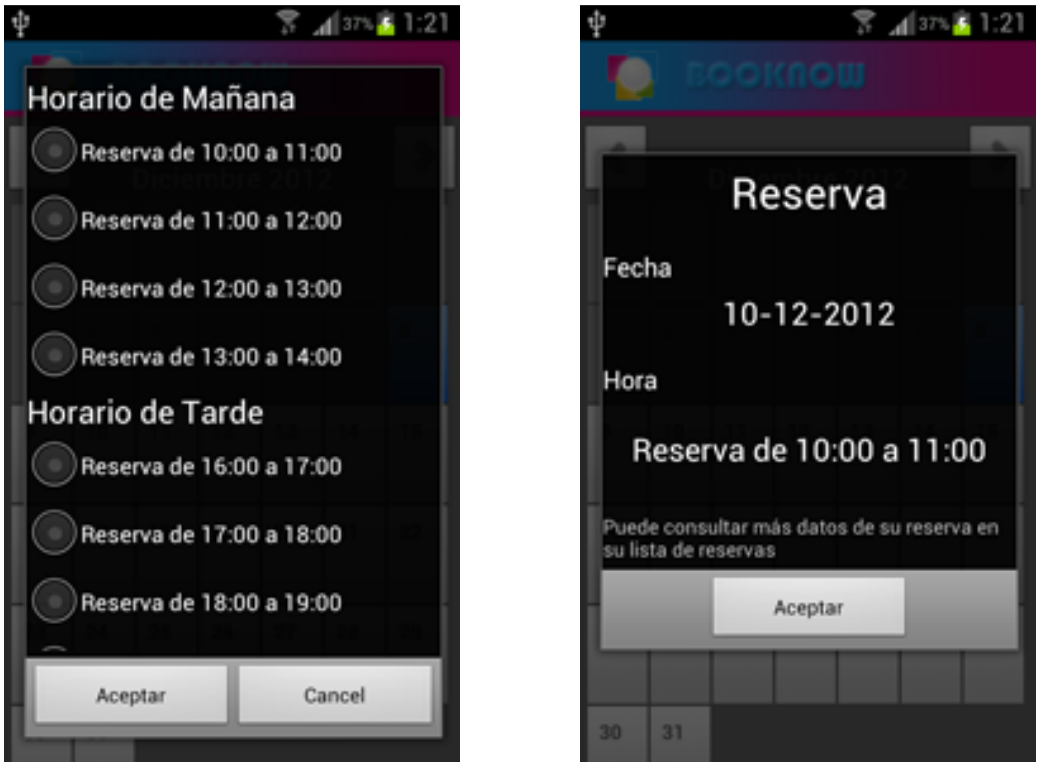


Ilustración 53. Pantalla de realización de reservas, cliente

Una vez terminada la reserva volvemos a la pantalla principal, ahora podemos consultar la reserva en el apartado Mis reservas, y pulsando sobre la reserva podemos verla detalladamente.

La reserva genera un código único que es el que tenemos que enseñar a la empresa el día que vayamos a realizar el servicio.

Ilustración 54. Pantalla de detalle de reservas, cliente

Desde la pantalla de detalle de reserva podemos ver la localización geográfica de la reserva, podemos realizar un recordatorio sobre esa reserva o eliminar la reserva.

Los recordatorios son avisos para que la aplicación nos avise la próxima vez que tengamos que acudir a un servicio, por ejemplo si realizamos una reserva para una peluquería y después queremos que la aplicación pasadas 4 semanas después de haber acudido a la reserva nos vuelva a recordar que pidamos hora con la peluquería realizaremos un recordatorio para 4 semanas, además podemos marcarlo como periódico de manera que cada 4 semanas se nos recordará que tenemos que ir a cortarnos el pelo a esa peluquería.

Cuando realicemos un recordatorio y pase el tiempo para que la aplicación nos avise aunque tengamos la aplicación apagada nos saltará una notificación en el móvil que nos avisará de la notificación y si la pulsamos se abrirá la aplicación mostrándonos la información del recordatorio.

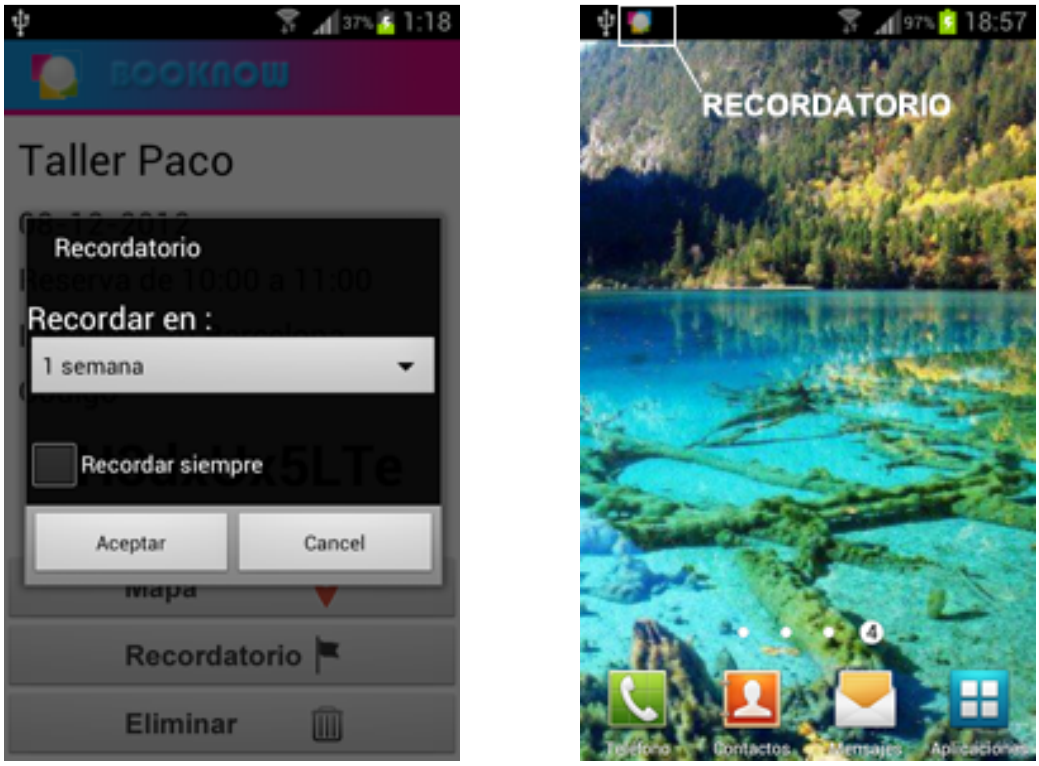


Ilustración 55. Pantallas de recordatorio, cliente

Una vez entramos en la aplicación nos avisará del recordatorio para el servicio que estaba relacionado y desde la pantalla de lista de reservar si pulsamos sobre el icono de recordatorio se mostrará un cuadro de diálogo con los recordatorios actuales.

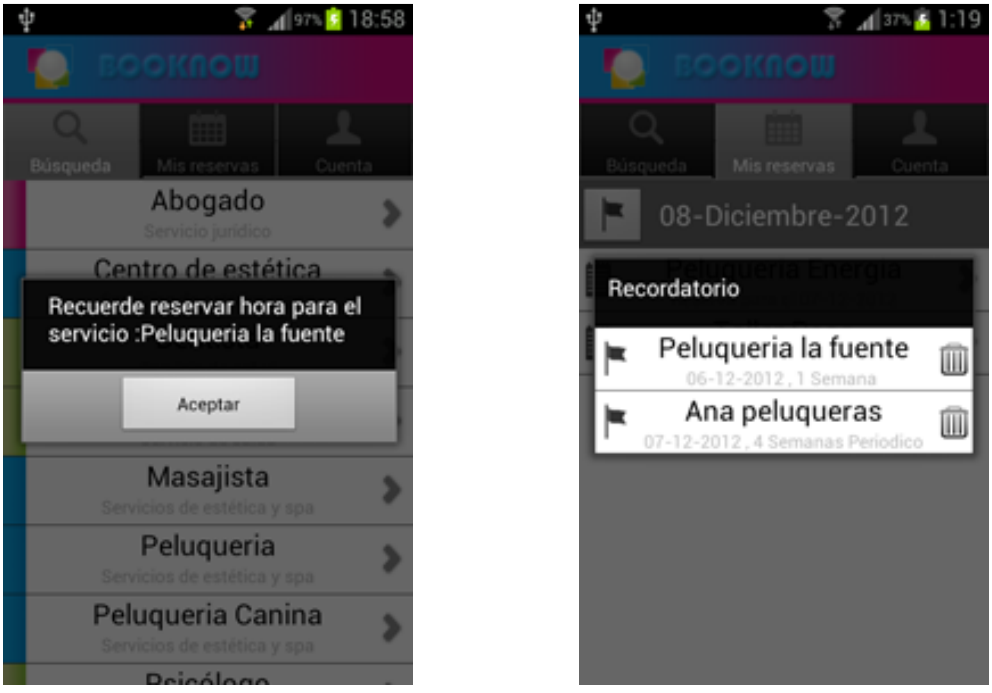


Ilustración 56. Pantallas de detalles de recordatorio, cliente

Si hacemos un pulsado largo sobre el recordatorio podemos eliminarlo, se nos mostrará un mensaje de aviso para confirmar la eliminación.

En el apartado de cuenta de usuario podemos realizar las mismas operaciones que en usuario empresa con la diferencia que no podemos editar la cuenta mientras tengamos reservas pendientes.

La aplicación mostrará diferentes mensajes para informar de lso diferentes eventos que se producen, uno de los mensajes a tener en cuenta es cuando no tengamos acceso a internet o problemas de conexión.

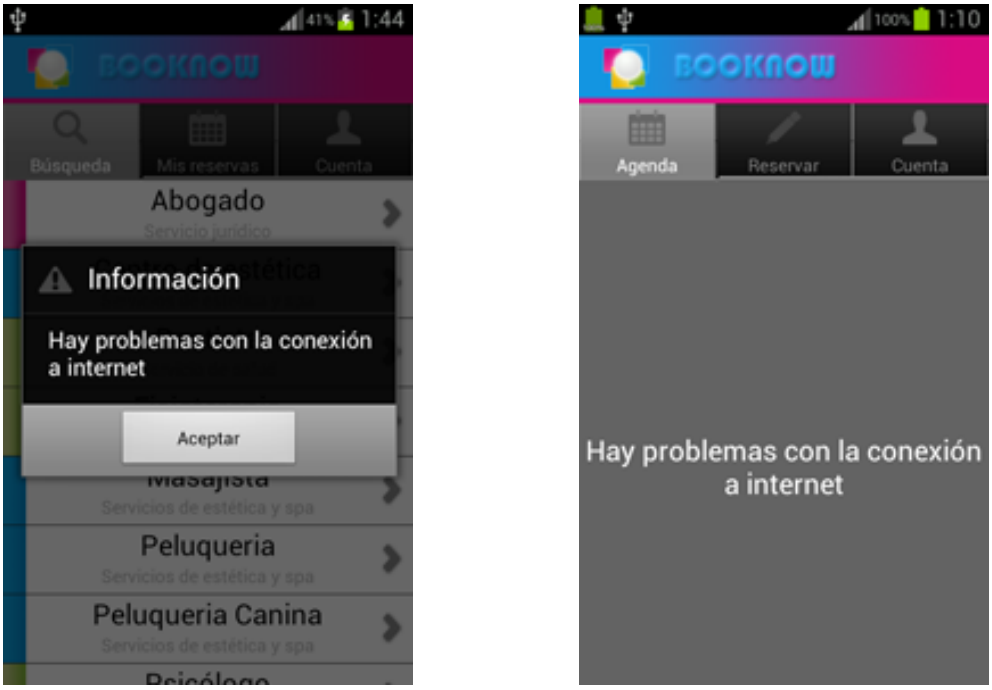


Ilustración 57. Pantallas de información de conexión