

Proyecto integrado de final de curso

Programa de gestión de hotel

Alumno: Rafael Suárez Franco

Grado Superior de Desarrollo de Aplicaciones Multiplataforma

I.E.S. Torre del Rey

Índice

1. - Introducción.....	4
2. - Estudio de Viabilidad.....	4
2.1. - Descripción del Sistema Actual.....	4
2.2. - Descripción del Sistema Nuevo.....	4
2.3. - Identificación de los Requisitos del Sistema.....	5
2.3.1. - Requisitos de Información.....	5
2.3.2. - Requisitos de Funciones.....	5
2.3.3. - Otros Requisitos.....	6
2.4. - Descripción de la Solución.....	8
2.5. - Planificación del Proyecto.....	9
2.6. - Estudio del Coste del Proyecto.....	10
3. - Análisis del Sistema de Información.....	10
3.1. - Identificación del Entorno Tecnológico.....	10
3.2. - Modelado de Datos.....	11
3.2.1. - Modelo Entidad-Relación.....	11
3.2.2. - Esquema de la Base de Datos.....	12
3.2.3. - Datos de Prueba.....	14
3.3. - Identificación de los Usuarios Participantes y Finales.....	15
3.4. - Identificación de Subsistemas de Análisis.....	15
3.5. - Establecimiento de Requisitos.....	16
3.6. - Diagramas de Análisis.....	21
3.7. - Definición de interfaces de usuario.....	24
3.7.1. - Especificación de principios generales de interfaz.....	25
3.7.2. - Especificación de formatos individuales de la interfaz de pantalla.....	27
3.7.3. - Especificación de formatos de impresión.....	38
3.7.4. - Especificación de la navegabilidad entre pantallas.....	43
4. - Construcción/Compilación del Sistema.....	43
4.1. - Acceso a datos.....	43
4.2. - Creación de componentes en tiempo de ejecución.....	44
4.3. - Visibilidad y compartimiento de información entre formularios.....	45
4.4. - Correos, encriptación, fechas y otros.....	46
5. - Breve manual de instalación y uso de la Aplicación.....	48
5.1. - Manual de instalación.....	48
5.2. - Manual de uso de de la aplicación (escritorio).....	49
5.2.1. - Registrarse.....	49
5.2.2. - Recuperar contraseña.....	49
5.2.3. - Visualizar y navegar entre fechas.....	49
5.2.4. - Reservar o administrar un día concreto.....	50
5.2.5. - Reservar o anular un periodo.....	51
5.2.6. - Visualizar todas mis reservas/ocupaciones.....	51
5.2.7. - Visualizar itinerario de servicios y otros informes y gráficos.....	51
5.2.8. - Crear habitaciones, servicios, etc.....	52
5.2.9. - Editar habitaciones, servicios, etc.....	53
5.2.10. - Preguntas frecuentes.....	53
5.3. - Manual de uso de de la aplicación (escritorio).....	54
5.3.1. - Registrarse.....	54
5.3.2. - Visualizar y navegar entre fechas.....	54
5.2.3. - Reservar o administrar un día concreto.....	54
5.2.4. - Reservar o anular un periodo.....	55
5.2.5. - Crear habitaciones, servicios, etc.....	55
6. - Conclusiones.....	56
7. - Bibliografía.....	58

1. - Introducción

Para este proyecto, desarrollaremos una aplicación de escritorio y una app móvil con funcionalidades similares. Dichas aplicaciones nos permitirán gestionar y hacer reservas en un hotel, ya sea desde el punto de vista de un administrador o de un cliente. En el programa podremos visualizar rápidamente el estado de las habitaciones, si están reservadas u ocupadas, los servicios que tienen contratados, etc. Los elementos del hotel (habitaciones, servicios, etc) serán ampliables, por lo que los componentes de la aplicación que representan a esos elementos también deben serlo. Por otra parte, necesitaremos almacenar los datos en una base de datos. Dado que queremos acceder desde distintos dispositivos, no contaremos con una BBDD local. Por último, también haremos uso de reports para imprimir información ordenada acerca del hotel (facturas, clientes, etc).

2. - Estudio de Viabilidad

2.1. - Descripción del Sistema Actual

Partimos de una aplicación bastante más sencilla, en la que contamos con un número fijo de habitaciones, servicios, etc. El programa originalmente está pensado para que un administrador se encargue de todas las reservas. Se hace uso de algún report, como un listado de los clientes.

2.2. - Descripción del Sistema Nuevo

A partir de la aplicación original, haremos que sea escalable para poder añadir habitaciones, servicios, temporadas, etc. Dichos elementos deben actualizarse en la interfaz, por lo que no se añadirán en el diseño predefinido de las pantallas, sino en tiempo de ejecución. Crearemos un sistema de usuarios para que los clientes puedan realizar sus propias reservas, añadiremos nuevos y más complejos informes y gráficos. Además, desarrollaremos la versión móvil de este programa.

2.3. - Identificación de los Requisitos del Sistema

2.3.1. - Requisitos de Información

La información que queremos guardar acerca del hotel es la siguiente. Véase el punto “3.2 Modelado de datos” para una explicación más exhaustiva de los datos que almacenaremos:

- Habitaciones: Guardaremos su número, precio y tipo descriptivo. Se estima que no habrán muchas habitaciones (10 para empezar).
- Clientes: Guardamos su información básica, nombre, apellidos, DNI, etc. El número de clientes guardados tampoco será demasiado alto.
- Usuarios: Contará con correo, nombre de usuario, contraseña, etc. Un cliente podrá tener o no tener un usuario asociado.
- Servicios: Tendremos su nombre y precio. No habrán muchos servicios, quizás entre 5 y 10.
- Reservas: Guardaremos las reservas y ocupaciones de cada habitación para cada fecha (día por día) además de los servicios y cliente de esa reserva concreta. Esta entidad será la que ocupará el mayor volumen de datos, dado que toda la información de la base converge en las reservas.
- Temporadas. Guardados la fecha o periodo de una temporada y su precio. No tendremos muchos registros de temporadas (unos pocos al año).

2.3.2. - Requisitos de Funciones

Dado que contaremos con dos tipos de perfiles, en primer lugar enumeraremos las posibilidades que tiene un usuario cliente, para más tarde presentar las opciones exclusivas de los administradores.

Un cliente podrá:

- Registrarse en el sistema y logearse. Será necesario iniciar sesión para acceder al resto del sistema, ya sea cliente o administrador. Las contraseñas se guardarán encriptadas en md5.
- Visualizar la pantalla principal, donde se encuentran las habitaciones disponibles. En dicha pantalla se puede ver el estado de todas las habitaciones en un día concreto.

- Navegar entre fechas. Para visualizar el estado de las habitaciones en otros días. Para los clientes es probable que esté restringido a fechas presentes y futuras.
- Reservar una habitación en un día concreto, o en un periodo de tiempo. El cliente podrá acceder al formulario de administración diario de cada día, siempre y cuando dicho día no esté reservado u ocupado por otro cliente. Las opciones del cliente se limitarán a elegir qué días y qué servicios reservar, será el administrador (repcionista) quien confirme las ocupaciones y tenga la postestad de manipular el precio y otros aspectos.
- Acceder al historial, presupuesto y facturas correspondientes al cliente.
- Anular reservas que estén a su nombre.
- Visualizar la pantalla mensual, para ver los estados futuros de las habitaciones de manera más generalizada. Permite visualizar las distintas temporadas establecidas por el hotel.

Un administrador podrá, además de todo lo anterior:

- Crear habitaciones, temporadas, servicios, clientes y usuarios. Podrá crear clientes sin usuarios, desde los formularios de reserva. Es decir, un cliente puede hacer una reserva y que la gestión la realice un administrador, en cuyo caso, puede introducir los datos del nuevo cliente a la vez que realiza la reserva.
- Informes: Crear facturas o presupuestos, ya sea por periodos para una habitación determinada o por clientes concretos. Imprimir otro tipo de informes, como itinerarios de servicio, historial de cliente, etc.
- Imprimir gráficos de las reservas u ocupaciones en un periodo concreto. Del mismo modo, imprimir un gráfico de la rentabilidad de los servicios del hotel.
- Administración total de las reservas y ocupaciones, capacidad de manipular precio y estado de las habitaciones (respetando unas reglas que enunciaremos a continuación), así como los servicios asociados dichas reservas.

2.3.3. - Otros Requisitos

Dado que estamos tratando con un proyecto de reservas, debemos establecer ciertas reglas a la hora de cambiar los estados de las habitaciones.

Reservar/anular periodos: solo se pueden hacer en días futuros, incluyendo el día de hoy. Anular también anula ocupaciones.

Reservas diarias (formulario diario):

- De libre a reservada : Día de hoy o futuro.
- De libre a ocupada : Día de hoy o en los 7 días pasados.
- De reservada a ocupada: Día de hoy o en el pasado.
- De reservada a libre : Cualquier día.
- De ocupada a libre : Día de hoy (o futuro, pero no se puede marcar como ocupado en el futuro).
- De ocupada a reservada: Nunca.

Reglas adicionales para usuarios clientes:

- Sólo se pueden manipular habitaciones futuras y que no estén reservadas por otros clientes. Si por ejemplo habíamos contratado servicios para hoy, ya no podemos echarnos atrás.
- Sólo pueden reservar, la ocupación la confirmará un administrador.

Por otra parte, aprovechamos este apartado para establecer el código de colores de la aplicación:

- Rojo: ocupado.
- Amarillo: reservado.
- Verde claro: libre - temporada baja.
- Verde: libre - temporada media.
- Verde oscuro: libre - temporada alta.

Los colores de las temporadas serán visibles desde la vista mensual, para saber aquellos periodos en los que las habitaciones estarán más caras. En cualquier caso mostraremos el color de reservas y ocupaciones por encima de las temporadas. Si una habitación tiene una reserva y pasa a estar ocupada, se mostrará el color rojo, aunque tengamos los dos registros.

2.4. - Descripción de la Solución

Para realizar el proyecto de reservas del hotel, utilizaremos Delphi 10.3 (lenguaje pascal), tanto para la versión de escritorio como para la versión móvil. Para la base de datos usaremos MySQL 5.5, dicha base estará alojada en una máquina virtual del instituto para ser accesible desde varios dispositivos. Otro software adicional que necesitaremos es Quick Report para construir informes y MyDAC, que permite trabajar con bases de datos MySQL en la versión de móvil (funciona como FireDAC). Dado que queremos tener un sistema de usuarios en el que los clientes se puedan registrar, haremos uso de protocolos para correos como SMTP y SSL.

Nos centramos en la programación por eventos, dado que el usuario interactúa con los componentes que hay en la interfaz gráfica. Debido a que las posibles opciones dependen del tipo de perfil del usuario, debemos contar con sesiones que controlen dicho perfil para determinar qué acciones se pueden realizar. Para este propósito, podemos hacer uso de variables y funciones/procedimientos públicos, es decir, serán visibles y accesibles en todo el programa. Esto nos permitirá pasar información entre distintas pantallas y ahorrar código, pudiendo contar con un formulario o archivo en el que situemos las funciones, tablas de BBDD, etc. a las que queramos acceder desde cualquier otra parte del programa. Otro aspecto a recalcar de este proyecto es que las características del hotel son ampliables, y por tanto, los componentes que representan las entidades de la base de datos también deben serlo. Esto se traduce en que algunas partes del programa no estarán predefinidas o construidas en su diseño inicial, sino que se generarán en tiempo de ejecución, dependiendo de los datos almacenados. Por ejemplo, si tenemos 10 habitaciones en principio pero vamos añadiendo otras nuevas, no nos servirá tener 10 botones en la pantalla, sino que debemos crear más.

En cuanto al lenguaje de base de datos, escogeremos MySQL, dado que es una opción familiar y asequible para la envergadura de este proyecto. Las aplicaciones móviles en Delphi normalmente están pensadas para el uso de BBDD internas con SQLite, pero dado que deseamos tener una base de datos común a la que podamos acceder desde distintos dispositivos, optaremos por contar con un servidor MySQL. En la app móvil necesitaremos MyDAC, un componente de terceros que permite trabajar con MySQL (homólogo de FireDAC en la app de escritorio). Para conectarnos y gestionar la base de datos con mayor facilidad y comodidad, utilizaremos el programa Heidi, desde el cual podemos visualizar los datos del servidor MySQL, crear y modificar las tablas, etc.

2.5. - Planificación del Proyecto

Equipo de trabajo:

Características del hardware de la máquina con la que desarrollaremos las aplicaciones:

- Procesador: AMD Ryzen 5 PRO 3400G 3.70 Ghz
- Memoria RAM: 16GB
- Gráfica: AMD Radeon Vega 11 Graphics.
- Sistema operativo Windows 10 64 bits.

Los requerimientos de Delphi son: Procesador de doble núcleo de 1,8 Ghz, 2GB RAM y windows 10, por lo que podremos trabajar con este software sin problemas.

En cuanto al servidor que alojará la base de datos, se trata de una máquina virtual Proxmox con las siguientes características: Ubuntu server, 2GB RAM, 20GB de almacenamiento, 2 núcleos de procesamiento.

Software necesario:

- MySQL 5.5
- Heidi 11.3
- Delphi 10.3 Rio
- QuickReport
- MyDAC

Ficheros necesarios (incluidos en el proyecto):

- Libmysql.dll
- Libeay.dll
- Ssleay.dll

2.6. - Estudio del Coste del Proyecto

Delphi es un programa cuya licencia es de pago. Sin embargo, cuenta con una versión 'community' gratuita para programas que no se comercien por cierta cantidad de dinero, por lo que podemos utilizarla en este caso.

Por otra parte, la mayoría de componentes de terceros para Delphi también son de pago, y algunos de ellos son necesarios para ciertas características que queremos implementar, como QuickReport o MyDAC. En cualquier caso, al ser éste un proyecto académico sin ánimo de lucro, haremos uso de pruebas gratuitas de los componentes mencionados.

En cuanto al resto del software, como MySQL, no hay ningún problema dado que también cuenta con versiones de community. En conclusión, no tendremos que desembolsar un solo euro para este proyecto, solo nuestro tiempo y esfuerzo.

3. - Análisis del Sistema de Información

3.1. - Identificación del Entorno Tecnológico

El entorno de Delphi es como cualquier otro entorno de desarrollo de software que podamos utilizar. Tanto QuickReport como FireDAC y MyDAC se integran y pueden ser utilizados dentro del entorno, es decir, al instalarlos, podemos disponer de los componentes para utilizarlos directamente en los formularios.

Por supuesto, contamos con una parte de diseño gráfico y otra parte para la codificación, pudiendo alterar el comportamiento de los componentes como deseemos. Cada pantalla de la aplicación será un formulario de Delphi (Unit1, Unit2, ...). Esto aplica en ambas versiones, y también a los reports (un formulario por cada report). Los formularios se pueden importar entre ellos para acceder a sus funciones y variables públicas, y por supuesto para que una pantalla pueda abrir otra pantalla distinta.

Por otra parte, usaremos Heidi para crear y gestionar la base de datos con comodidad, dado que podemos crear las tablas y datos de manera visual, sin necesidad de SQL. Simplemente tenemos que crear una conexión con la máquina donde queramos crear la

base de datos. De esta forma podemos visualizar los datos para comprobar que la aplicación funciona como deseamos.

3.2. - Modelado de Datos

3.2.1. - Modelo Entidad-Relación

- Descripción de necesidades (breve explicación o enunciado de los datos que queremos guardar):

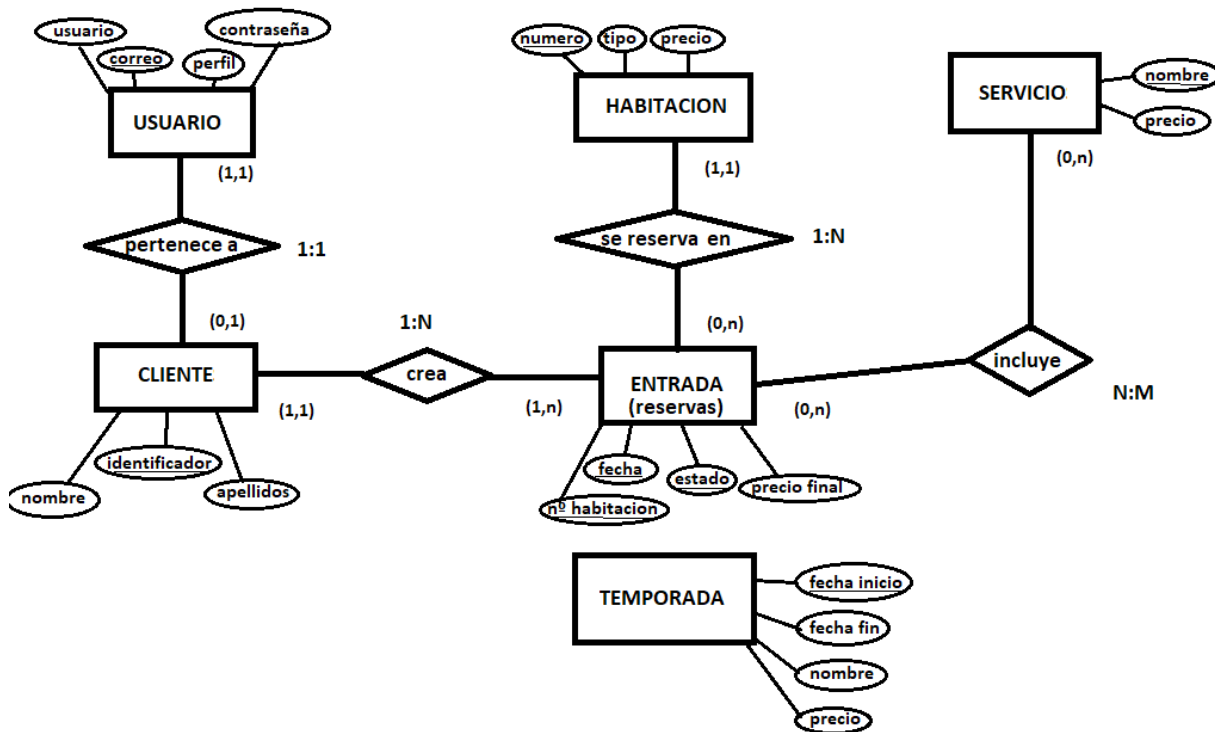
Necesitamos guardar las reservas y ocupaciones que se realizan en el hotel, de las cuales queremos guardar el nº de habitación, la fecha, el estado (si es reserva u ocupación), el cliente titular de la entrada y el precio que pagará o pagó (ya sea reserva u ocupación). Si una habitación pasa de reservada a ocupada, nos interesa guardar el registro de que estuvo previamente reservada (es decir, no modificamos o borramos la reserva, en ese caso guardamos los registros con los dos estados distintos). De las habitaciones, guardaremos su número, tipo y precio unitario por día. Una habitación puede estar reservada y/u ocupada un mismo día, como ya hemos dicho. Del cliente guardaremos el nombre, apellidos y un identificador (DNI por ejemplo) de momento nos ahorraremos el teléfono u otros datos personales. Los clientes pueden tener o no un usuario para esta aplicación (si no tiene, un administrador gestionará sus reservas y ocupaciones). Del usuario guardaremos un correo, nombre de usuario, contraseña y tipo de perfil (usuario o admin). Por otra parte, el hotel cuenta con servicios que se pueden contratar individualmente para cada día. De los servicios guardaremos el nombre y el precio. Debemos guardar qué servicios se contratan para qué días y qué habitaciones. Por último, hay que tener en cuenta las temporadas. Dependiendo de en qué fecha se reserva o se ocupa, el precio puede subir (temporada media o baja). Las temporadas las creará el administrador manualmente, dado que pueden variar de un año a otro. Nos interesa guardar su fecha de inicio y de fin, el precio adicional, y un nombre descriptivo (media, baja, alta, etc).

- Descripción del model E/R:

Como podemos ver en la imagen del modelo Entidad Relación, la entidad o tabla principal de este proyecto será la de entradas, que recogerá las reservas y ocupaciones del hotel. su clave principal será el número de habitación junto con la fecha y el estado, dado que una habitación en un mismo día puede estar reservada y más tarde ocupada. La

relación con las habitaciones es de uno a varios, dado que una habitación tendrá 0 o varias entradas, pero cada entrada es de una sola habitación. En cuanto a los clientes, la relación es la misma que con las habitaciones (no contamos con que hayan varias personas ocupando la habitación, solo tenemos en cuenta el titular de la reserva). Por lo que su clave principal se guardará en entradas para tener una referencia al cliente. Por otra parte, como ya hemos dicho, los clientes pueden tener un usuario, aunque no es obligatorio (el administrador puede dar de alta un cliente pero no tiene por qué crearle un usuario en el sistema) Por lo que la clave del cliente pasará también a la tabla de usuarios. En cuanto a los servicios, hay una relación de varios a varios dado que en una entrada se pueden contratar varios servicios y un mismo servicio puede estar contratado en varias entradas. En cuanto a las temporadas, no es necesario crear ninguna relación con alguna entidad del modelo, dado que simplemente usaremos esta tabla para consultar fechas en ellas. Su clave principal puede ser, por ejemplo, la fecha de inicio, dado que no debe repetirse (las temporadas no pueden solaparse en sus periodos).

MODELO E/R HOTEL



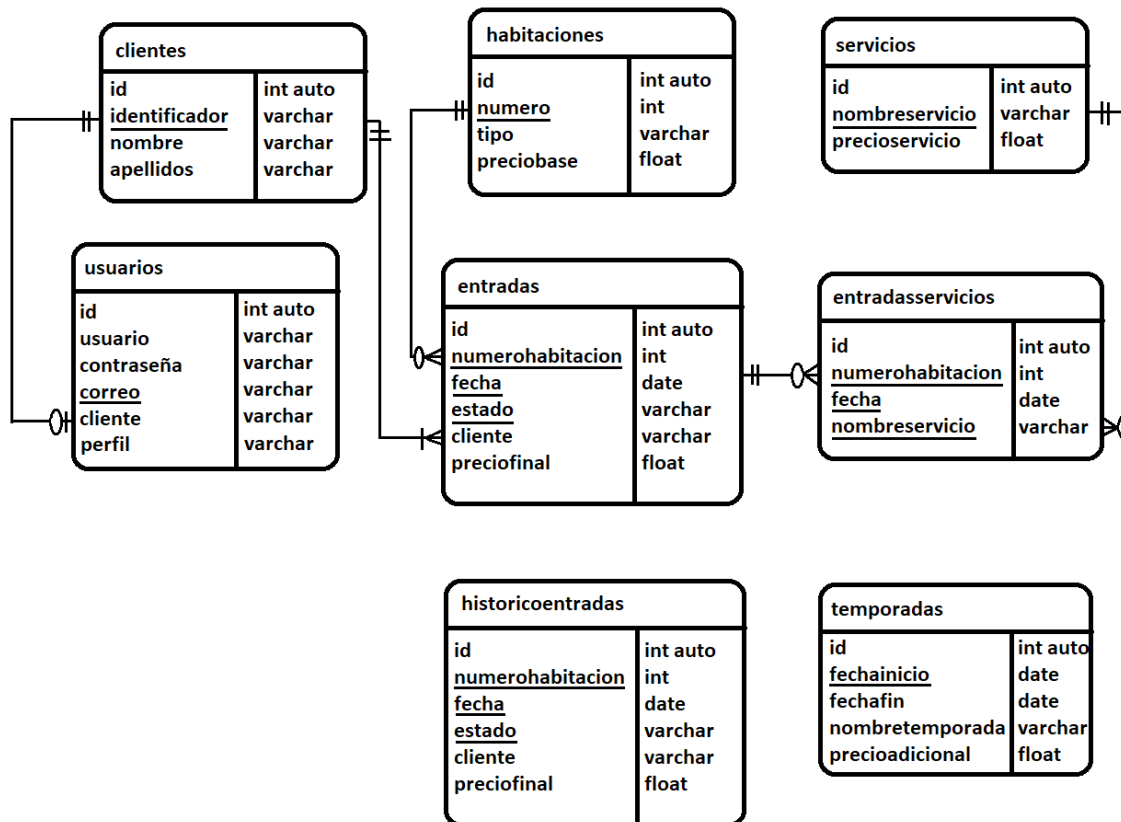
3.2.2. - Esquema de la Base de Datos

- Modelo patas de gallo:

Por último para tener una vista de las tablas que vamos a crear y utilizar, podemos hacer uso de un diagrama de patas de gallo. Cada entidad que hemos mencionado tendrá su

propia tabla. La única relación que generará una tabla nueva será la que hay entre entradas y servicios, que nombraremos "entradasyservicios", la cual se compone de las claves primarias de las otras dos. Sin embargo no contamos con el estado de la tabla entradas. Se entiende que los servicios contratados, por ejemplo en una reserva, serán los que se prestarán si esa reserva se convierte en ocupación. Si en el programa es necesario diferenciar entre servicios contratados (reserva) y servicios prestados (ocupación) siempre se puede hacer un join a la tabla entradas y comprobar el estado. Por otra parte la clave de habitaciones se propaga a entradas, para pasar a ser parte de la clave principal de dicha tabla. Sin embargo la clave "identificador" de clientes se queda con el rol de clave foránea, dado que no es necesaria para diferenciar las distintas reservas/ocupaciones. Como hemos dicho, consideramos que una reserva/ocupación tiene un único cliente. Hay que destacar que, dado que no guardamos cuando una habitación está libre, se entiende que si dicha habitación para un día concreto no tiene registro en la tabla, se considera libre. Esto quiere decir que anular una reserva u ocupación implica eliminar registros de la tabla entradas, dado que no consideramos un tipo de estado "libre" o "liberado". Para no perder información sobre las anulaciones, tendremos una tabla (historicoentradas) idéntica a la tabla de entradas en la que guardaremos todos los registros borrados de dicha tabla. La misma lógica podría aplicar a la tabla entradasyservicios, pero consideramos que guardar los servicios que se han cancelado no es tan relevante. En cualquier caso esta es la planificación inicial de la base de datos del programa, por lo que no estamos cerrados a hacer cambios. En cuanto a la tabla de temporadas, como hemos dicho, solo la usaremos para consultar fechas en sus registros, no hay relación directa con otras tablas. Su lógica es parecida a la de la tabla entradas: si una fecha no cae en un registro, quiere decir que cae en temporada baja, y que por tanto, no se sube el precio.

HOTELRSF MODELO DE PATAS DE GALLO



3.2.3. - Datos de Prueba

Como datos de pruebas, tendremos una cantidad moderada de registros en la mayoría de tablas, por ejemplo, unas 10 habitaciones, 5 servicios, 4 temporadas, 5 clientes, 2 usuarios. En cuanto a las entradas y entradas-servicios, tendremos bastantes más, en el orden de los 100 o 200.

Los datos se encontrarán en el servidor MySQL, el cual será accesible siempre y cuando tengamos conexión a internet. En cualquier caso, se adjuntará un fichero SQL que contendrá la base de datos y algunos datos de prueba, en caso de que fuera necesario. También se adjuntará un fichero con las credenciales del servidor, para que en caso de que no esté en línea o el servicio se haya detenido, podamos reiniciarlo manualmente.

3.3. - Identificación de los Usuarios Participantes y Finales

Como hemos indicado anteriormente, tendremos 2 tipos de usuarios: administradores y clientes, que serán diferenciados según su campo “perfil” en la tabla de usuarios del sistema.

El rol de administrador va dirigido a los empleados del hotel, ya sea al recepcionista, o cualquier otra persona que tenga que gestionar las reservas de dicho hotel. En cualquier caso, dicho rol estará diseñado para tener todo tipo de privilegios en la gestión del hotel, dado que consideramos que el hotel en cuestión no será de gran tamaño, y que por tanto podría estar gestionado por una sola persona.

En cuanto a los clientes, sus posibilidades obviamente estarán más restringidas. No podrán crear habitaciones o servicios, ni acceder a reservas de otros clientes, etc. Ya hemos explicado más detalladamente las opciones de ambos tipos de usuarios en el apartado “2.3.2 Requisitos de Funciones” por lo que no nos extenderemos más en este punto.

3.4. - Identificación de Subsistemas de Análisis

En Delphi, como en muchos otros programas de aplicaciones de escritorio, trabajamos con formularios que representan las distintas pantallas a las que podemos acceder en la aplicación final. A priori se podría decir que cada formulario podría representar un subsistema o contener una funcionalidad específica del programa, pero ese no será nuestro caso, dado que varios formularios pueden cumplir algunas funciones que pertenezcan a un mismo subsistema, o puede que en una misma pantalla, se accedan a funcionalidades de diferentes subdivisiones. Los subsistemas o principales partes en las que se divide este programa son los siguientes:

- Navegación y visualización de fechas y reservas: será la parte principal y que estará más a la vista del usuario, dado lo primero que hay que tener en cuenta a la hora de reservar es ver o consultar los días o periodos en los que cada habitación está libre.
- Reserva/ocupación de habitaciones: estrechamente relacionado con lo anterior. Una vez determinamos cuándo podemos reservar, podemos proceder a realizar reservas u ocupaciones. Aquí podemos incluir también la reserva de servicios.
- Creación/edición de las entidades del hotel: Ya sea crear habitaciones, servicios,

temporadas, etc u editarlas. Todas aquellas funciones que amplíen la capacidad y opciones del hotel, y que por tanto no serán utilizadas tan frecuentemente.

- Informes: Tendremos la opción de imprimir informes o listados con la información del hotel, además de ciertos gráficos sobre la evolución del mismo.
- Usuarios: En todo momento la aplicación debe saber qué tipo de usuario está logeado, en especial si es un cliente, dado que el comportamiento y accesibilidad del programa variará en función de ese factor.
- Bases de datos: Podemos considerar que hay otro subsistema que se encarga del acceso y operaciones en bases de datos, y puede que sea el más importante dado que todas las funcionalidades del programa hacen consultas o cambios en la BD.

3.5. - Establecimiento de Requisitos

En este apartado vamos a relacionar los subsistemas descritos con los requisitos funcionales del punto 2.3.2.

En primer lugar, el subsistema de navegación/visualización de reservas: En este subsistema incluimos los requisitos de funciones siguientes: Visualizar la pantalla principal, Visualizar la pantalla mensual y Navegar entre fechas. Hacemos uso de dos formularios, "Principal" y "PantallaMes" (Units 1 y 2) con los que podemos visualizar las reservas de las habitaciones en distintas fechas. en Principal podemos ver todas las habitaciones existentes representadas como paneles, las cuales se colorean dependiendo de su estado y de las temporadas.

En esta pantalla podemos navegar entre fechas para consultar los estados de las habitaciones en días concretos. Si queremos visualizar las reservas de una habitación para un determinado mes, pulsamos en el botón "Abrir Mes" de la habitación en cuestión. Esto nos llevará a PantallaMes, donde obtenemos una vista generalizada de las reservas. En esta pantalla también podemos navegar entre fechas (de mes en mes) y cambiar la habitación que queremos consultar.

En ambos formularios se crean los paneles en tiempo de ejecución, dado que no tenemos un número de habitaciones fijo, y los meses del año varían también. Además, hay que recolorear cada vez que cambiamos de fecha para actualizar los estados.

Subsistema de reservas. Incluimos las funciones tanto de reserva como de anular habitaciones. Para este apartado tenemos las pantallas FormularioPeriodo y

FormularioDiario. Tal como sus nombres indican, permiten reservar un periodo o un día concreto. La opción de anular reservas está contemplada en los mencionados formularios.

En el caso de los administradores, contamos con un formulario más, el de AltaCliente, dado que puede ser que estén realizado una reserva para un cliente nuevo y lo tengan que dar de alta durante la operación. Esta opción no estará disponible para los clientes, dado que al estar registrados, ya tuvieron que dar de alta su registro de cliente previamente, y al estar logeados, se utiliza su identificador automáticamente como titular de la reserva.

Para acceder al FormularioDiario, podemos hacer click derecho en una habitación de la pantalla Principal y pulsar Administrar, o bien ir a su PantallaMes y hacer click en el día que queramos. Tengamos en cuenta que un cliente no podrá abrir el formulario para una habitación que esté reservada por otro cliente. Para reservar, ocupar o liberar la habitación, se hará uso de los radio botones del formulario, aunque prevalecerán las reglas establecidas anteriormente.

En cuanto al FormularioPeriodo, se puede acceder desde el Principal. Tendremos 2 botones para abrirlo en modo reserva o anulación. Las opciones son similares a las del anterior formulario, salvo que tendremos que especificar un periodo. Si un periodo para una habitación contiene registros, no se podrá realizar la reserva. Un cliente podrá anular cierto periodo arbitrariamente, sin que esto afecte a reservas de otros clientes. Un administrador en cambio podrá anular cualquier registro.

En ambas pantallas tendremos un pequeño panel que contendrá los servicios existentes en el hotel. El usuario puede seleccionar aquellos que desee. En el caso del periodo, se contratarán los servicios seleccionados por cada día del periodo. En cualquier caso, se puede acceder posteriormente a un día concreto para cambiar los servicios, si se desea. El precio de los servicios se añade o resta automáticamente al precio final de la habitación. Recordamos que sólo el administrador puede modificar ese precio en última instancia.

Subsistema de creación y edición de las entidades del hotel. Aquí incluimos aquellos formularios que permiten nuevos elementos del hotel. En concreto tenemos pantallas para crear habitaciones, servicios, temporadas y usuarios. También tenemos de alta de clientes, pero ese caso lo consideramos una excepción dado que nunca tendremos la necesidad de crear un cliente si no es para registrar una reserva o un usuario nuevo. En

cuanto al de usuarios, podría decirse que es parte del subsistema de usuarios, dado que va dirigido a que los clientes puedan registrarse, pero incluimos la opción aquí también dado que un administrador podría crear otros admins o clientes.

En cada uno de los formularios nos aseguramos de que la clave principal no se repita, es decir, no queremos habitaciones o servicios con el mismo número o nombre. El caso más especial es el de las temporadas, dado que tenemos que comprobar que un nuevo periodo no se cruce con otro que existiera anteriormente, por lo que tenemos que asegurarnos mediante código de que así sea. Debido a que trabajamos con Delphi, contamos con algunos componentes específicos que se relacionan directamente con la base de datos, con sus tablas y en concreto, con sus campos, por lo que podemos tener entradas de texto por ejemplo que solo admitan números si el tipo de campo es integer. Lo explicaremos más en detalle en el subsistema de DB. En otros casos utilizaremos componentes normales y tendremos que restringir el input del usuario mediante código.

En cuanto a la edición de elementos, de manera similar a las reservas, reutilizaremos los mismos formularios, los cuales tendrán un "modo edición". Esta opción está pensada para cambiar los atributos de los elementos, como sus precios. No tendremos una opción de eliminar elementos, dado que no consideramos que sea necesario en un proyecto de esta extensión. Más allá de borrar las reservas y servicios anulados, no hay necesidad de eliminar otro tipo de datos. Las opciones de este subsistema serán accesibles desde la barra de menú en la pantalla Principal.

Subsistema de informes y gráficos: Tendremos acceso a gráficos y distintos reports que nos ofrecerán información en forma de listados. Estas opciones también serán accesibles desde la barra de menú de Principal.

Tenemos 5 tipos de de informes. En primer lugar, Factura o presupuesto. Imprime una lista de reservas u ocupaciones para un periodo y habitación determinados. Se entiende que las ocupaciones son las que se factura y las reservas las que se presupuestan, dado que no se han cobrado todavía. Un cliente puede ver este informe, pero solo incluirá los registros a su nombre. El resto de informes a continuación solo son para administradores. Tenemos un listado de clientes, que muestra toda la información de los clientes existentes, un historial de cliente, similar a factura, pero filtrado por cliente y no por periodo. Un itinerario de servicios, que muestra los servicios contratados para cada habitación en un día concreto. Para seleccionar el día del que queremos el itinerario, debemos navegar a ese día en la pantalla Principal. Por último, tenemos la opción de imprimir un listado de cualquier tabla de la base de datos. Contaremos con un pequeño formulario que nos permite seleccionar qué tabla, sus campos y campo para ordenar.

Este formulario puede servir, por ejemplo, para ver el histórico de reservas anuladas, las habitaciones existentes, los correos de los usuarios, etc.

En cuanto a los gráficos tenemos 2. En cuanto a código, son parecidos a los informes, dado que debemos construir una o varias consultas para obtener la información que mostraremos. El primero es un gráfico de barras que muestra las reservas u ocupaciones en un periodo. Dichas barras serán de color amarillo o rojo, de acuerdo con el código de color establecido. En el mismo gráfico tenemos otra parte que muestra en barras azules los ingresos de esas reservas/ocupaciones (sumatorios de los precios finales de esos registros). En segundo lugar tenemos un gráfico circular o de sectores que muestra la proporción de servicios contratados en cierto periodo y para cierta habitación (o para todas las habitaciones), para reserva u ocupación. También muestra el total de ingresos conseguidos o esperados por esos servicios, por lo que este gráfico nos permite ver nuestros servicios más rentables o populares, o analizar en qué habitaciones son más propensos a ser contratados.

Subsistema de usuarios: En este subsistema incluimos todo aquello relacionado con los usuarios, como el formulario de Login y de alta de usuario, pero también aquel código en otros formularios que controla las acciones posibles, dependiendo del tipo de usuario logeado. Por lo tanto, este subsistema está presente en todo el programa.

En cuanto al Login, es la primera pantalla al acceder a la aplicación. Si bien la Principal figura como primer formulario, el Login se activa y muestra dentro del evento "onActivate" del Principal, por lo que de ese modo, requerimos que el usuario se registre antes de acceder al resto de la aplicación. Esto lo hacemos así para que al cerrar el Principal, no nos devuelva al Login. Si lo que queremos es cambiar de usuario, contaremos con un botón de "logout" que relanzará la aplicación, mostrando de nuevo el Login. Dentro de login podemos introducir las credenciales para acceder al sistema, o podemos registrarnos, lo cual nos llevará al formulario de creación de usuario. Por supuesto no nos permitirá crear un administrador, pero podremos crear el cliente si no existe (del mismo modo que lo haría un administrador al hacer una reserva). Para completar el registro, tendremos que introducir un código generado aleatoriamente que se envía al correo que hemos indicado. Para enviar correos en una app Delphi, necesitamos 3 componentes: Un socket SSL, un IdSMTP y un IdMessage. También necesitamos una cuenta de correo que haga de remitente, la cual hemos creado previamente, y los ficheros ssleay.dll y libeay.dll. Por último tenemos la opción de recuperar contraseña. Del mismo modo, verifica el usuario mediante un código que se envía a su correo. Si el código es correcto, se le permite cambiar su contraseña con unos inputboxes.

Ya hemos explicado las restricciones que tienen los usuarios clientes. Para aplicar esas restricciones, en todo momento guardamos el tipo de perfil, el identificador del cliente y su nombre de usuario para saber quién está logeado. Esa información se mostrará en la pantalla Principal, junto al botón de logout. Una vez tenemos esa información, en cada uno de los eventos active de los formularios tendremos las restricciones que consideremos oportunas. En el propio active del principal, tras obtener la info del usuario, realizamos una serie de acciones. Por ejemplo, quitar del menú principal las opciones de crear habitaciones, servicios, etc. Si intentamos entrar en el FormularioDiario de una habitación reservada por otro cliente, en su active se analiza si el cliente coincide o no. En caso negativo, se cierra el formulario. En caso positivo, el campo de cliente del formulario se rellena automáticamente con el identificador del cliente (y se pone en modo de sólo lectura). Lo mismo aplica para el formulario de periodo, y en cuanto al informe de factura o presupuesto, se filtra automáticamente por cliente. Esas serían, en resumen, las restricciones de las que se encarga este subsistema.

Subsistema de bases de datos: Similar al anterior, este subsistema se encuentra en todo el programa, dado que en cualquier parte del mismo debemos hacer consultas y registros en base de datos. Dado que el proyecto y la BD tienen una dimensión considerable, contaremos con un formulario en el que situaremos los componentes relacionados con bases de datos, al cual hemos llamado "Tablas". Esto quiere decir que si queremos acceder a una de las tablas o queries desde otro formulario, tenemos que importar y llamar a Tablas para realizar la acción que queramos.

En Tablas tendremos una conexión a la BD (elemento FireDAC: FDConnection, en android sería MyDAC: MyConnection) y otros elementos, FDTable (MyTable en android), uno por cada tabla del sistema. Debemos abrir dichas tablas antes de poder realizar operaciones con ellas. Opcionalmente, una FDTable puede tener un DataSource, que sirve para mostrar información en otros elementos. Por ejemplo, hemos mencionado que al crear un elemento, una habitación, podemos tener campos de texto relacionados con campos de una tabla. Para esa conexión necesitaríamos un DataSource de la FDTable en cuestión. Por otra parte, al añadir los campos de una FDTable, se crean componentes que representan esos campos, por ejemplo "FDTableentradasfecha" Es el campo fecha de la FDTable de entradas. Mediante dichos campos podemos acceder al valor en el que se sitúa el puntero de una determinada FDTable, y ya sea para consultarlo, modificarlo, eliminarlo, etc. Por otra parte tenemos las FDQuerys (MyQuery) que permiten hacer consultas en lenguaje SQL. Son útiles si las consultas son complejas y largas. En este proyecto tendremos una FDQuery para las consultas que vayamos necesitando (la sobreescribimos siempre) y otras que utilizaremos en los

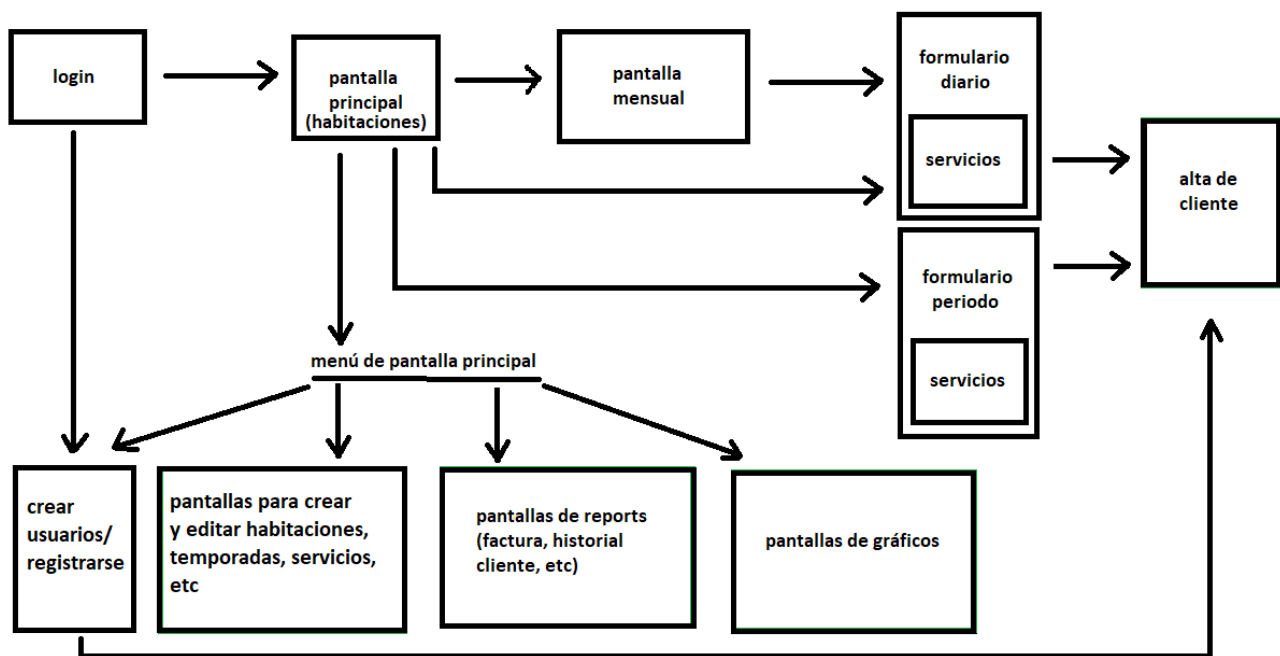
informes y gráficos. Para poder usar los componentes FireDAC, necesitamos la libmysql.dll, incluida en este proyecto.

Por último cabe mencionar que dado que Tablas es un formulario que importaremos en todas o casi todas las demás pantallas, podemos situar en él aquellas funciones que queramos usar en distintas partes del programa. Basta con incluir la Unit del formulario en la sección "uses" (ejemplo: uses Unit3;).

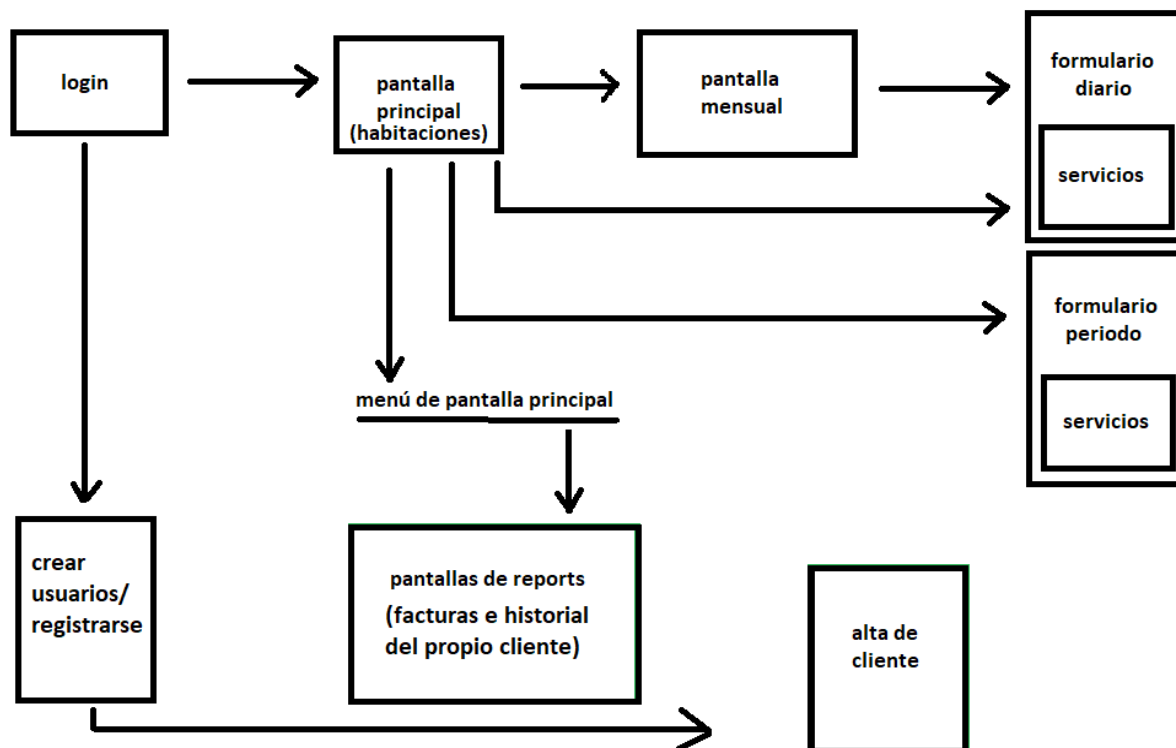
3.6. - Diagramas de Análisis

A continuación, mostraremos unos esquemas simples de las pantallas que formarán parte del programa. Podemos diferenciar entre 4 variantes, dependiendo de si estamos en escritorio o móvil, y de si somos administrador o no.

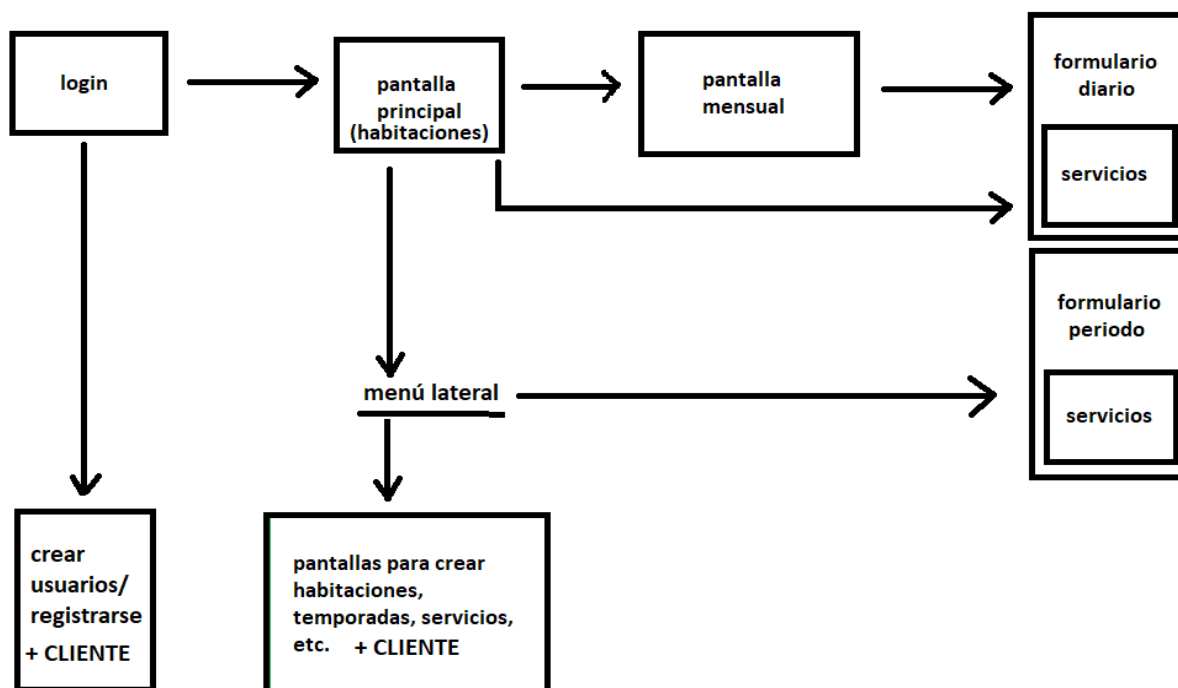
Administrador en escritorio.



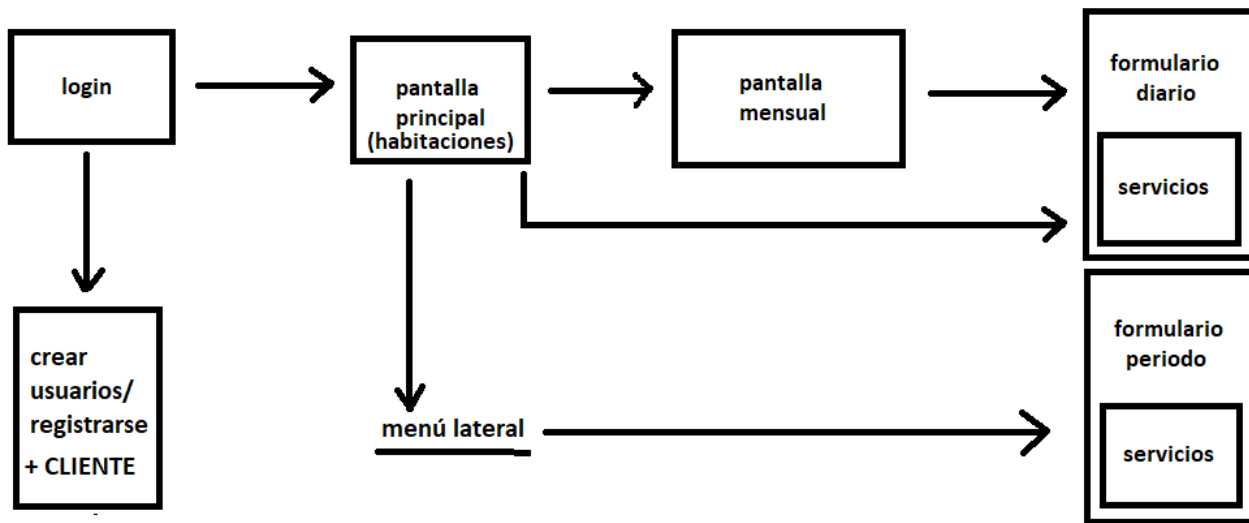
Cliente en escritorio.



Administrador en móvil.

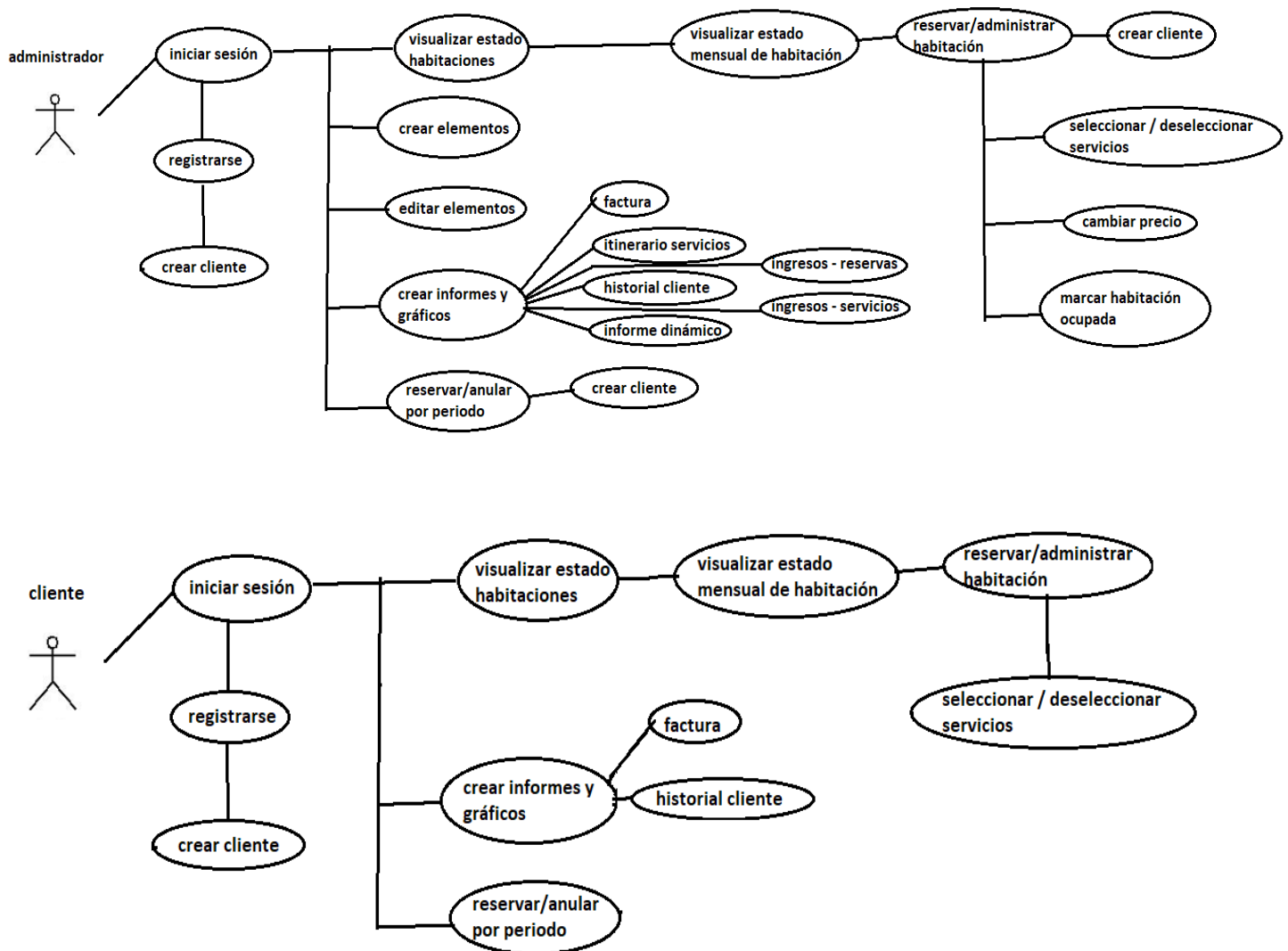


Cliente en móvil.



Como hemos explicado anteriormente, la versión de escritorio cuenta con más características (y por tanto, pantallas) que la móvil. De la misma manera, un cliente tiene el acceso restringido a ciertos formularios o funcionalidades. Las diferencias más importantes que caben destacar son que en la versión android no hacemos uso de una pantalla “modal” para crear clientes, debido a las limitaciones del desarrollo para dicha versión. De hecho, el concepto de pantalla modal no se utiliza o no está permitido en delphi android como tal, por lo que debemos crear el cliente a parte. Por otra parte, la mayoría de funciones del menú en escritorio no están disponibles para la versión móvil. Hemos incluido las de crear elementos (habitaciones, clientes) para que haya algo de diferencia entre un cliente y un administrador. Las opciones de editar en este caso no las incluimos en android, dado que no son muy importantes y redundan bastante. En cualquier caso mantenemos la estructura de pantalla principal y mensual junto con los dos formularios de reserva, que son la parte principal del programa y la que se utilizará con mayor frecuencia.

Por otra parte, en los siguientes diagramas de casos de uso mostramos de forma simplificada las funcionalidades que hemos descrito anteriormente, ya sea para administrador o para cliente.



3.7. - Definición de interfaces de usuario

3.7.1.- Especificación de principios generales de interfaz

- Libertad y control del usuario: Sobre todo aplica cuando hablamos de un usuario administrador, el cual tiene la capacidad de acceder a cualquier función del sistema. Para un cliente, esa libertad estará más acotada, por ejemplo, no podrá acceder a una habitación que esté ya reservada. Aún así, dentro de esos límites o reglas, también se respeta su libertad de decisión. Además, la mayoría de decisiones son reversibles, se pueden anular reservas de habitaciones y/o servicios. No es posible borrar habitaciones, clientes, etc ya creados, dado que no queremos borrar información de la base de datos, y peligraría la integridad referencial (si borramos un servicio que ya ha sido contratado, ¿qué hacemos con sus registros?).

- Visibilidad del estado del sistema: La aplicación normalmente informará al usuario del resultado de sus acciones mediante cajas de diálogo o mensajes, para que sepa si se ha ejecutado correctamente o no la acción que estaba realizando. Además, algunas acciones tendrán efectos visibles en la interfaz. Por ejemplo, cambiar el estado de las habitaciones cambiará sus colores en la pantalla principal o mensual.

- Correspondencia entre el sistema y el mundo real: En este programa, la información se presenta de manera ordenada, sobre todo el plano temporal. Por ejemplo, la pantalla mensual muestra las reservas y ocupaciones que hay en cada día de cierto mes, por lo que es fácil ver qué días están libres, en qué día de la semana caen, etc. Por otra parte, el hecho de que exista una habitación o un servicio “real” al que representan las habitaciones y servicios de la aplicación ya queda bajo la responsabilidad de los administradores del hotel, pero obviamente recomendamos que no se creen habitaciones que no existen, dado que no consideramos la opción de borrarlas y no tiene sentido que un cliente las pueda reservar.

- Diseño estético y minimalista: El diseño de la app es bastante sencillo. En la versión escritorio tiene un tema oscuro con tonalidades verdes. Delphi nos da la opción de usar temas predefinidos para facilitar este apartado. No obstante, en la versión móvil, no es tan sencillo (además de que los temas suelen ser de pago) por lo que hemos optado por un tema más claro con colores suaves, dado habría que cambiar todos los componentes para darle un aspecto similar a la otra versión. Por otra parte, no hay una gran cantidad de controles o componentes en cada pantalla para no saturar al usuario, y como hemos explicado, se hace uso de colores para transmitir información de manera simplificada e intuitiva (verde = libre, rojo = ocupado, similar a un semáforo).

- Coherencia y estándares: A este respecto no hay mucho más que decir que no hayamos

dicho en puntos anteriores. Mencionamos de nuevo el uso de colores, y en general, el procedimiento o comportamiento de cada pantalla es similar por lo que respetamos cierta coherencia.

- Prevención de errores: Durante el desarrollo de este proyecto hemos tenido en cuenta los posibles errores que puedan surgir, según los inputs que pueda introducir el usuario, por lo que idealmente estarán controlados en el código. Por ejemplo, intentar crear un cliente que ya existe, o crear una temporada donde la fecha de fin sea anterior a la de inicio. Gran parte del código de cada formulario se dedica a impedir que ese tipo de errores se lleven a cabo, y la mayoría de ellos también devolverán cierta retroalimentación para informar al usuario de qué está haciendo mal. También hacemos uso de bloques try – except para controlar partes de código que pueden causar que el programa se cuelgue o que ocasionalmente fallen, como por ejemplo, el envío de correos electrónicos.

- Ayuda y documentación: Más adelante en este documento, facilitaremos un manual de uso de la aplicación.

- Reconocimiento en vez de recordar: Normalmente, el usuario no tendrá necesidad de recordar ciertas instrucciones o acciones sobre la aplicación, dada su baja complejidad. En cualquier caso, y reiterando en puntos anteriores, en caso de que el usuario se equivoque, normalmente recibirá un mensaje que le guiará para que realice la acción correctamente. A parte de esto, todas las opciones y funcionalidades del programa están a la vista, por lo que no debería haber problema a la hora querer utilizar una u otra.

- Flexibilidad y eficiencia de uso: La aplicación puede ser usada por todo tipo de usuarios. En general, está más enfocada a usuarios con poca experiencia o que necesiten un poco de guía, pero también cuenta con aspectos que van dirigidos a usuarios experimentados, como el uso de atajos de teclado.

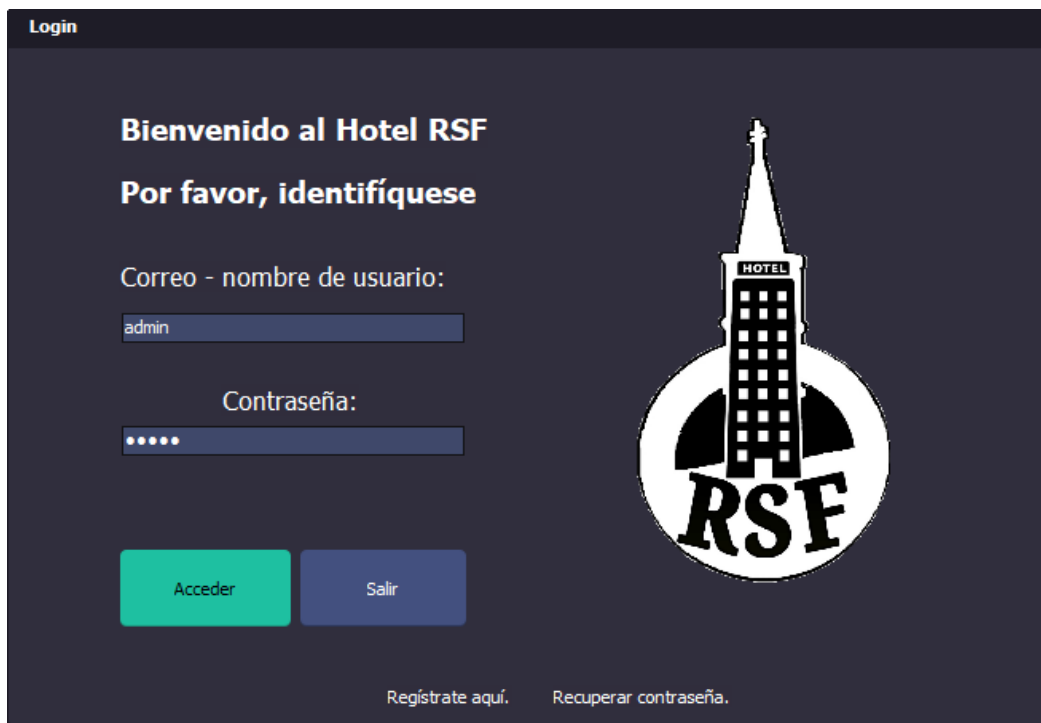
- Ayuda a los usuarios a reconocer, diagnosticar y recuperarse de errores: Como ya hemos explicado, en cualquier caso que se produzca un error, se mostrará un mensaje indicando lo que se está haciendo mal, y lo que el usuario debería hacer para corregir el error.

3.7.2.- Especificación de formatos individuales de la interfaz de pantalla

Antes de mostrar los diseños de las pantallas, indicamos que cada una de ellas tiene unas dimensiones fijas que no se pueden cambiar, para evitar que se oculten los controles si el usuario decide cambiar el tamaño. Por tanto, además, normalmente prescindiremos de los botones de maximizar, minimizar, e incluso el de cerrar, dependiendo de lo que queramos en cada formulario. Cada formulario aparecerá en el centro de la pantalla, por defecto. El nombre de cada pantalla aparece en la barra superior (en escritorio), y es el mismo nombre que le hemos dado en el código (algunos de ellos ya los hemos mencionado), por ejemplo, aquellos que empiezan por la palabra “Alta” son los que sirven para crear algún elemento. En cualquier caso, se puede cambiar el nombre que aparece en esa barra por otro distinto. En este punto mostraremos también las diferencias que puede haber en una misma pantalla, dependiendo del tipo de usuario u otros factores.

En primer lugar, mostraremos las pantallas de escritorio.

Login :



The screenshot shows a login interface for 'Hotel RSF'. The title bar at the top left says 'Login'. The main content area has a dark background. On the left, the text 'Bienvenido al Hotel RSF' is displayed in white, followed by 'Por favor, identifíquese'. Below this, there are two input fields: 'Correo - nombre de usuario:' with the value 'admin' and 'Contraseña:' with five dots. At the bottom left, there are two buttons: 'Acceder' (green) and 'Salir' (blue). At the bottom right, there are two links: 'Regístrate aquí.' and 'Recuperar contraseña.'. On the right side of the screen, there is a large white logo of a hotel building with 'HOTEL' written above it and 'RSF' written below it in a stylized font.

Principal (admin):

Principal

Crear Editar Informes Gráficos

Fecha actual: 06/06/2022 Fecha seleccionada: 06 06 2022

Volver al día de hoy

Saltar al día: junio de 2022

do.	lu.	ma.	mi.	ju.	vi.	sá.
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2
3	4	5	6	7	8	9

Día anterior

Reservar periodo

Anular periodo

Logout

Usuario: admin
Perfil: admin
Cliente:

Abrir mes

Habitación 1 (simple)

Abrir mes

Habitación 2 (simple)

Abrir mes

Habitación 3 (simple)

Abrir mes

Habitación 4 (doble)

Abrir mes

Habitación 5 (doble)

Abrir mes

Habitación 6 (doble)

Abrir mes

Habitación 7 (doble)

Abrir mes

Habitación 8 (doble)

Abrir mes

Habitación 9 (doble)

Abrir mes

Habitación 10 (suite)

Abrir mes

Habitación 34 (suite)

Abrir mes

Habitación 12 (Ultra cara)

Principal (cliente): Podemos ver a simple vista que tiene menos opciones de menú.

Principal

Informes

Factura

Historial por cliente

Fecha actual: 06/06/2022 Fecha seleccionada: 06 06 2022

Volver al día de hoy

Saltar al día: junio de 2022

do.	lu.	ma.	mi.	ju.	vi.	sá.
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	1	2
3	4	5	6	7	8	9

Día anterior

Reservar periodo

Anular periodo

Logout

Usuario: cliente
Perfil: cliente
Cliente: 1234

Abrir mes

Habitación 1 (simple)

Abrir mes

Habitación 2 (simple)

Abrir mes

Habitación 3 (simple)

Abrir mes

Habitación 4 (doble)

Abrir mes

Habitación 5 (doble)

Abrir mes

Habitación 6 (doble)

Abrir mes

Habitación 7 (doble)

Abrir mes

Habitación 8 (doble)

Abrir mes

Habitación 9 (doble)

Abrir mes

Habitación 10 (suite)

Abrir mes

Habitación 34 (suite)

Abrir mes

Habitación 12 (Ultra cara)

PantallaMes:

PantallaMes

Fecha seleccionada: 06/06/2022 Habitación seleccionada: 1

Salto al día: 06 06 2022

Cambiar habitación: 1

Lunes	Martes	Miércoles	Jueves	Viernes	Sábado	Domingo
		Día 1	Día 2	Día 3	Día 4	Día 5
Mes anterior	Día 6	Día 7	Día 8	Día 9	Día 10	Día 11
	Día 12	Día 13	Día 14	Día 15	Día 16	Día 17
	Día 18	Día 19	Día 20	Día 21	Día 22	Día 23
	Día 24	Día 25	Día 26	Día 27	Día 28	Día 29
	Día 30	Día 31				

Mes siguiente

FormularioDiario (el panel de servicios no aparece hasta que la habitación se reserva):

FormularioDiario

Habitación: 3

Fecha: 18/02/2022

Ciente: 12345678A

Precio actual: 21

Estado

- ☐ libre
- ☐ reservada
- ☒ ocupada

Servicios

- ☒ Lavandería (4€)
- ☒ Desayuno (7€)
- ☐ Limpieza (5€)

Precio final: 21

Registrar

FormularioDiario

Habitación: 1

Fecha: 07/06/2022

Ciente:

Precio actual: 10

Estado

- ☒ libre
- ☐ reservada
- ☐ ocupada

Registrar

FormularioPeriodo (modo reserva – modo anular):

The image displays two side-by-side screenshots of a web application titled 'FormularioPeriodo'. The left screenshot shows the 'Reservar' (Reserve) mode, and the right screenshot shows the 'Anular' (Cancel) mode.

FormularioPeriodo (modo reserva – modo anular):

FormularioPeriodo (modo reserva):

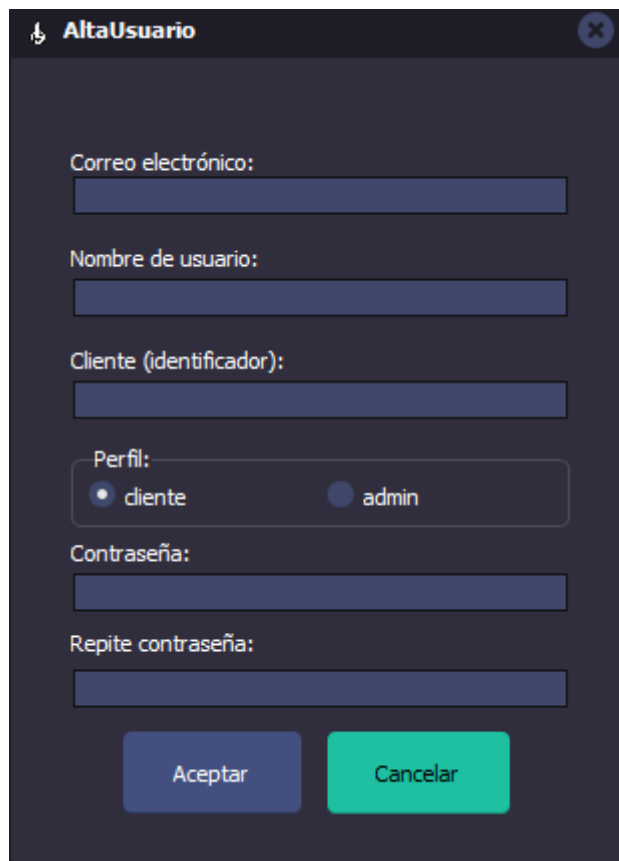
- Fecha inicio: 18 02 2022
- Fecha fin: 24 02 2022
- Habitación: 4
- Cliente: 12345678A
- Servicios:
 - ☐ Lavandería (4€ al día)
 - ☒ Desayuno (7€ al día)
 - ☒ Limpieza (5€ al día)
- Reservar

FormularioPeriodo (modo anular):

- Fecha inicio: 06 06 2022
- Fecha fin: 06 06 2022
- Habitación: 1
- Anular

En el modo anular, prescindimos del cliente y los servicios, dado que anularemos registros independientemente de ellos.

AltaUsuario: Este formulario puede ser llamando desde Login o desde Crear > Usuario (si somos admin). La única diferencia es que desde Login no se permite seleccionar el perfil de administrador.



A dark-themed modal window titled "AltaUsuario" with a close button in the top right corner. The form contains several input fields and a radio button group. The fields are: "Correo electrónico:" (empty), "Nombre de usuario:" (empty), "Cliente (identificador):" (empty), "Perfil:" with two radio buttons labeled "cliente" (selected) and "admin", "Contraseña:" (empty), and "Repite contraseña:" (empty). At the bottom are two buttons: "Aceptar" (blue) and "Cancelar" (green).

AltaUsuario

Correo electrónico:

Nombre de usuario:

Cliente (identificador):

Perfil:

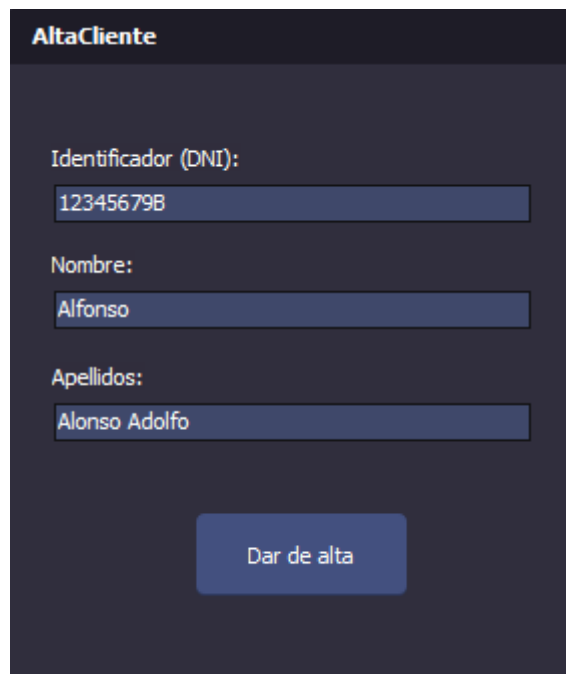
☒ cliente ☐ admin

Contraseña:

Repite contraseña:

Aceptar Cancelar

AltaCliente:



A dark-themed modal window titled "AltaCliente". The form contains three input fields: "Identificador (DNI):" with the value "12345679B", "Nombre:" with the value "Alfonso", and "Apellidos:" with the value "Alonso Adolfo". At the bottom is a single button labeled "Dar de alta".

AltaCliente

Identificador (DNI):

12345679B

Nombre:

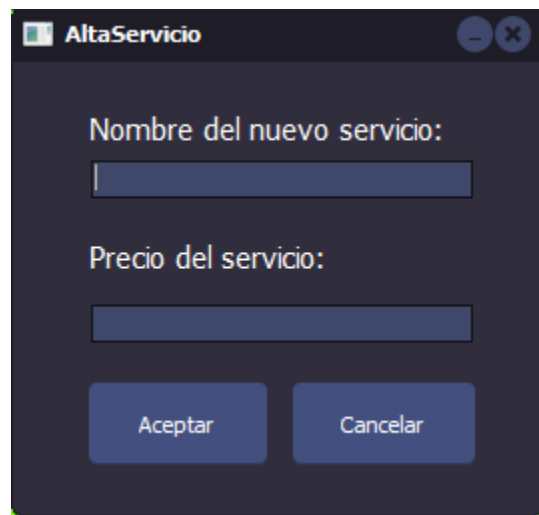
Alfonso

Apellidos:

Alonso Adolfo

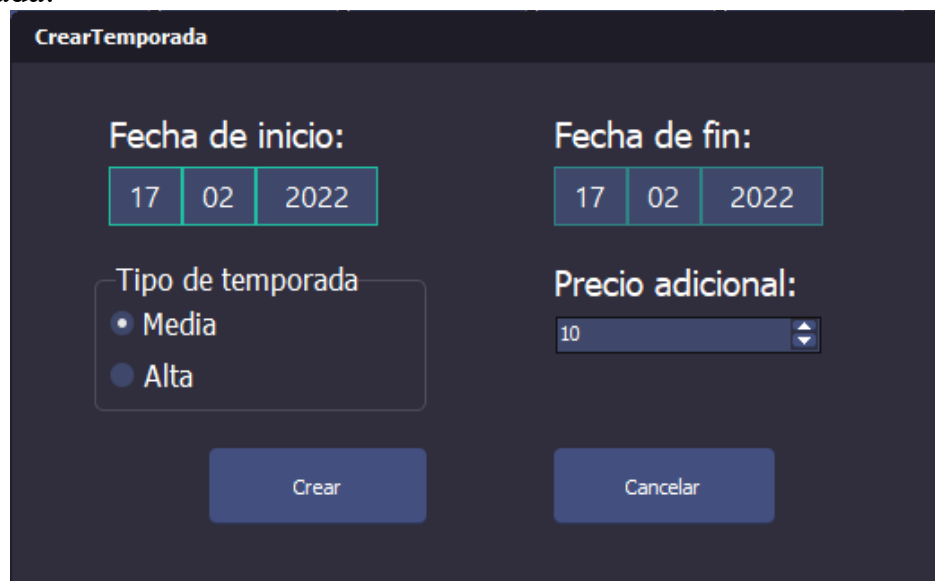
Dar de alta

AltaServicio:



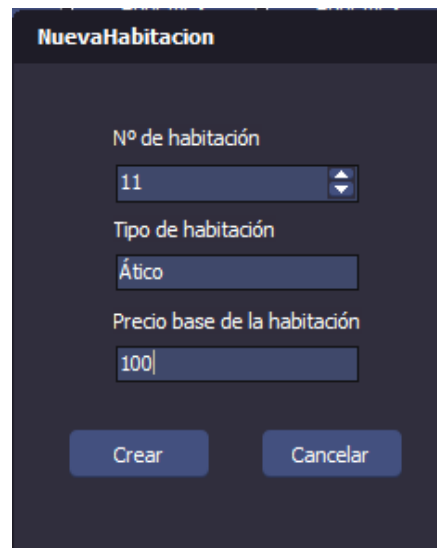
A screenshot of a software dialog box titled "AltaServicio". It has a dark background with light-colored text. At the top, there is a title bar with a small icon and the text "AltaServicio", followed by standard window control buttons (minimize, maximize, close). The main area contains two labels: "Nombre del nuevo servicio:" followed by a text input field, and "Precio del servicio:" followed by another text input field. At the bottom, there are two buttons: "Aceptar" and "Cancelar".

AltaTemporada:



A screenshot of a software dialog box titled "CrearTemporada". It has a dark background with light-colored text. The title bar shows the text "CrearTemporada". The main area is divided into four sections: "Fecha de inicio:" with a date picker showing "17 02 2022"; "Fecha de fin:" with a date picker showing "17 02 2022"; "Tipo de temporada:" with two radio buttons, "Media" (selected) and "Alta"; and "Precio adicional:" with a numeric input field showing "10". At the bottom, there are two buttons: "Crear" and "Cancelar".

AltaHabitacion:



A screenshot of a software dialog box titled "NuevaHabitacion". It has a dark background with light-colored text. The title bar shows the text "NuevaHabitacion". The main area contains three labels: "Nº de habitación" with a numeric input field showing "11"; "Tipo de habitación" with a text input field showing "Ático"; and "Precio base de la habitación" with a numeric input field showing "100". At the bottom, there are two buttons: "Crear" and "Cancelar".

Respecto a las pantallas de edición, son idénticas a las pantallas de “Alta” que acabamos de mostrar, por lo que no es necesario que enseñemos más capturas. De hecho solo hay dos diferencias: El texto del botón “Crear” se cambia por “Editar”, y antes de mostrar el formulario, se pide por inputbox la clave primaria del elemento a cambiar (por ejemplo, el nº de habitación) para asegurar primero que el elemento a editar existe.

Las pantallas de informes y gráficos las dejamos para el siguiente apartado, sin embargo, contamos con otras dos pantallas que sirven para pasar parámetros a algunos de esos informes, por lo que las mostramos en este punto también.

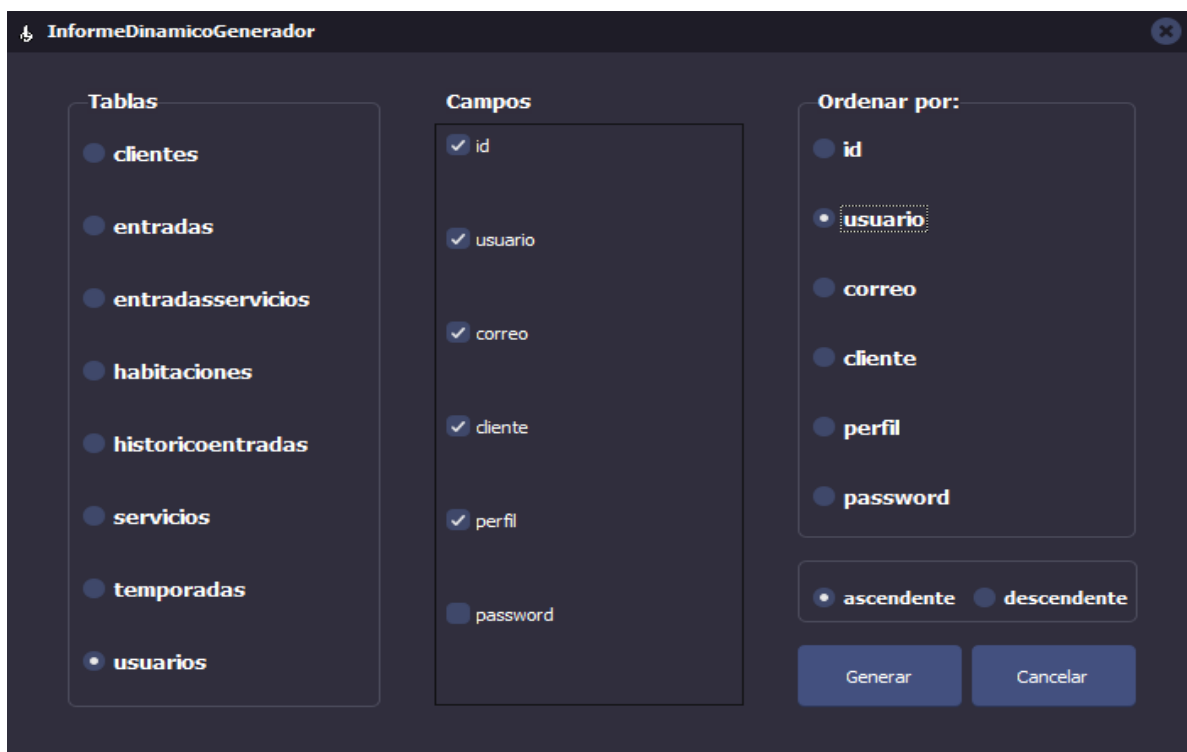
FacturaParametros: Para el informe de “Factura”, seleccionamos habitación, periodo, y reservas u ocupaciones.



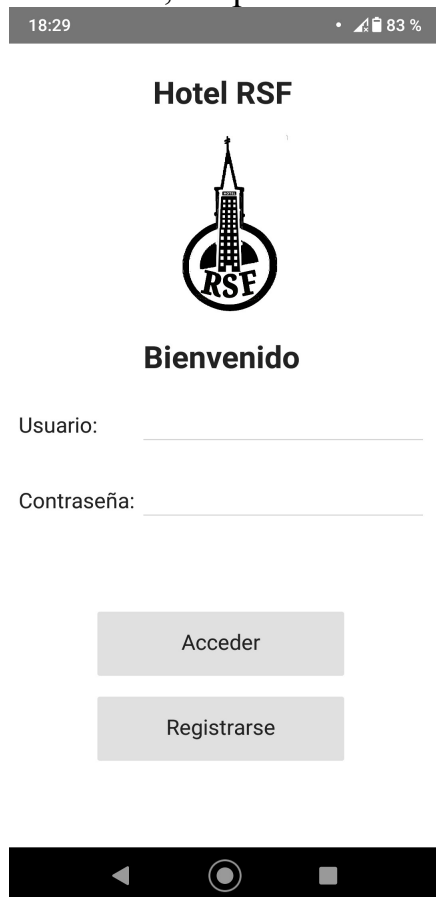
The screenshot shows a web form titled "FacturaParametros" with a dark blue background. It contains the following fields and controls:

- Fecha inicio:** A date picker showing 06/06/2022.
- Fecha fin:** A date picker showing 06/06/2022.
- Habitación:** A dropdown menu with the value "1" selected.
- Estado:** A group box containing two radio buttons: "reservas" (which is selected) and "ocupaciones".
- Buttons:** Two buttons at the bottom: "Generar factura" and "Cancelar".

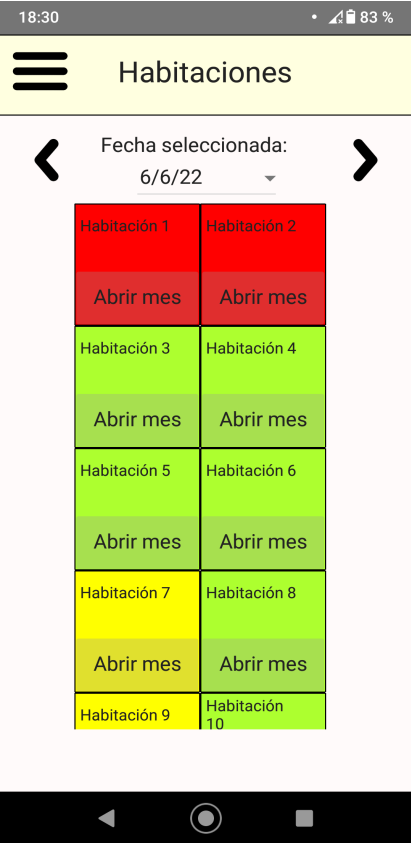
InformeDinamicoGenerador: Permite escoger la tabla de BD de la que queremos informe, sus campos, y un campo para ordenar.



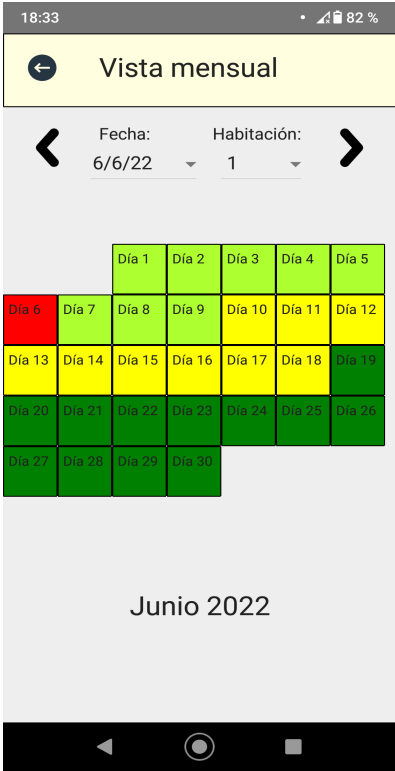
A continuación, las pantallas de la versión android. Login y registro de usuario:



Pantalla principal y menú lateral (admin y cliente):



PantallaMes:



FormularioDiario:

18:33 82 %

← Administrar habitación

Habitación: 1 Estado:

☒ Libre

Fecha: 10/6/22 ☐ Reservada

☐ Ocupada

Cliente: _____

Precio: 10

Aceptar

18:33 82 %

← Administrar habitación

Habitación: 1 Estado:

☐ Libre

Fecha: 10/6/22 ☒ Reservada

☐ Ocupada

Cliente: 12345

Precio: 34

Servicios	Precio final:
<input checked="" type="checkbox"/> Lavandería (3€)	34
<input checked="" type="checkbox"/> Desayuno (7€)	
<input type="checkbox"/> Limpieza (5€)	
<input type="checkbox"/> Masajes (4,4€)	

Aceptar

FormularioPeriodo (reserva – anular):

18:32 82 %

← Reservar periodo

Fecha inicio: 6/6/22

Fecha fin: 6/6/22

Habitación: 1

Cliente: _____

Servicios:

- ☐ Lavandería (3€ al día)
- ☐ Desayuno (7€ al día)
- ☐ Limpieza (5€ al día)
- ☐ Masajes (4,4€ al día)
- ☐ Bar (4,5€ al día)

Reservar

18:32 82 %

← Anular periodo

Fecha inicio: 6/6/22

Fecha fin: 6/6/22

Habitación: 1

Cliente: _____

Anular

Pantallas para crear elementos:

18:33 82 %

← Nuevo Cliente

Identificador (DNI):

Nombre:

Apellidos:

Dar de alta

18:33 82 %

← Nueva Habitación

Nº de Habitación

< 0 >

Precio Base

Tipo de Habitación

Crear Habitación

14:09 61 %

← Nuevo Servicio

Nombre del servicio:

Precio del servicio:

Dar de alta

18:33 82 %

← Nueva Temporada

Fecha de inicio 6/6/22

Fecha de fin 6/6/22

Tipo ☒ Media ☐ Alta

Precio adicional < 0 >

Crear Temporada

3.7.3.- Especificación de formatos de impresión

A continuación, mostraremos los diseños de los documentos impresos que podemos generar con este programa.

Listado de clientes:

The screenshot shows a report design window titled 'InformeClientes'. The design is laid out on a grid with columns numbered 1 to 20 and rows numbered 1 to 7. The report structure is as follows:

- Row 1:** Title band containing the text 'LISTADO DE CLIENTES'.
- Row 2:** Header band containing three fields: '{IDENTIFICADOR CLIENTE}', '{APELLIDOS}', and '{NOMBRE}'.
- Row 3:** Detail band containing three fields: '{dentificado}', '{apellidos}', and '{nombre}'.
- Row 4:** Footer band containing three fields: '{A fecha de: {Date/Time}}', '{Total clientes: {Detail count}}', and '{Página: {Page#}}'.

Un informe sencillo, muestra todos los clientes de la BD (en la banda blanca de detalle). Podemos aplicarle un acabado de 'cebra' en el código.

LISTADO DE CLIENTES		
IDENTIFICADOR CLIENTE	APELLIDOS	NOMBRE
12345678C	Alonso Adolfo	Alfonso
1234	De confianza	Mi cliente
12345	El de los palotes	Pepito
12345678B	Pepero Pepaso	Pepe
123456	q	q
12345678A	Suárez Franco	Rafael

Factura e historial de cliente: Ambas opciones utilizan el report de 'Factura' (dado que el historial de cliente es una factura filtrada por su identificador). Este report cuenta con subreports o subbandas para mostrar los servicios contratados cierto día. Distinguimos entre reservas y ocupaciones dado que lo lógico es que hayamos cobrado los días que realmente se han ocupado. En cualquier caso, para seguir el código de colores, las reservas van en amarillo y las ocupaciones en rojo.

Factura																			
1	FACTURA - INFORME POR PERIODO																		
2	Del día o cliente																		
3																			
4																			
5																			
6																			
7																			
8																			
9																			
10																			

Ejemplo de factura:

FACTURA - INFORME POR PERIODO				
Del día 05/01/2022 al 08/05/2022				
FECHA	Nº HABITACIÓN	ESTADO	ID CLIENTE	PRECIO DEL DÍA
12/02/2022	1	reservada	1234	10
Sin servicios contratados				
23/02/2022	1	reservada	1234	21
Sin servicios contratados				
SERVICIOS CONTRATADOS		PRECIO DEL SERVICIO		
Lavandería		4		
Desayuno		7		
25/02/2022	1	reservada	1234	10

Ejemplo de historial de cliente:

HISTORIAL DE CLIENTE				
Del cliente: Pepito El de los palotes				
FECHA	Nº HABITACIÓN	ESTADO	ID CLIENTE	PRECIO DEL DÍA
02/02/2022	34	reservada	12345	50
Sin servicios contratados				
04/02/2022	9	ocupada	12345	20
Sin servicios contratados				
04/02/2022	2	reservada	12345	14
SERVICIOS CONTRATADOS		PRECIO DEL SERVICIO		
Desayuno		7		
05/02/2022	2	reservada	12345	10

Mediante código, podemos sustituir el subreport por una etiqueta que indique si cierto día no hay servicios contratados.

Itinerario de servicios: Similar al anterior, muestra los servicios que se deben prestar cierto día.

ItinerarioServicios

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

1

2

3

4

5

6

7

8

9

10

ITINERARIO DE SERVICIOS

Del día 01-01-2000

NOMBRE DEL SERVICIO: nombre

HABITACIONES

Habitación numero

A fecha de: {Date/Time}

Página: {Page#}

Ejemplo:

ITINERARIO DE SERVICIOS

Del día 05/03/2022

NOMBRE DEL SERVICIO: Lavandería

HABITACIONES

Habitación 5

Habitación 14

Habitación 34

NOMBRE DEL SERVICIO: Desayuno

Sin habitaciones para este día.

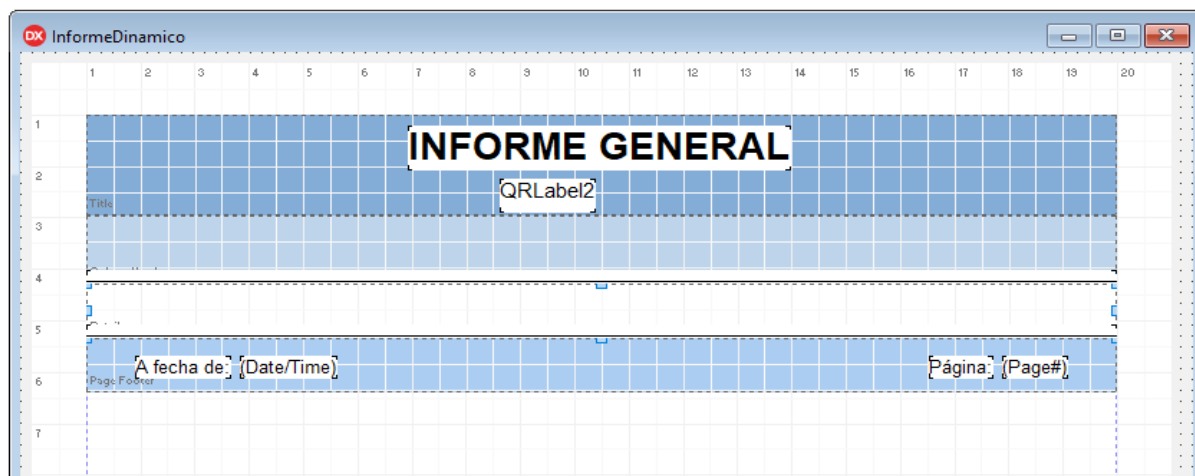
NOMBRE DEL SERVICIO: Limpieza

HABITACIONES

Habitación 5

Habitación 14

Informe dinámico: Como podemos ver, las bandas de este informe están vacías en principio, dado que las crearemos y posicionaremos según los campos que hayamos elegido en la pantalla que vimos en el punto anterior.



Ejemplo: Todos los servicios ordenados por precio de mayor a menor.

INFORME GENERAL	
De la tabla servicios	
NOMBRE SERVICIO	PRECIO SERVICIO
Desayuno	7
Limpieza	5
Bar	4,5
Masajes	4,4
Lavandería	3

O todas las habitaciones, ordenadas por su tipo.

INFORME GENERAL		
De la tabla habitaciones		
NUMERO	TIPO	PRECIO BASE
4	doble	20
5	doble	20
6	doble	20
7	doble	20
8	doble	20
9	doble	20
1	simple	10
2	simple	10
3	simple	10
10	suite	50
34	suite	50
12	Ultra cara	1112

En cuanto a los gráficos, sus diseños son predefinidos, lo único que varía son las variables que ponemos en cada eje y la consulta, por lo que simplemente mostraremos un par de ejemplos.

Gráfico de reservas: muestra las reservas e ingresos de esas reservas para el mes de febrero (cada barra azul son los ingresos a la izq de sus reservas, barra amarilla).

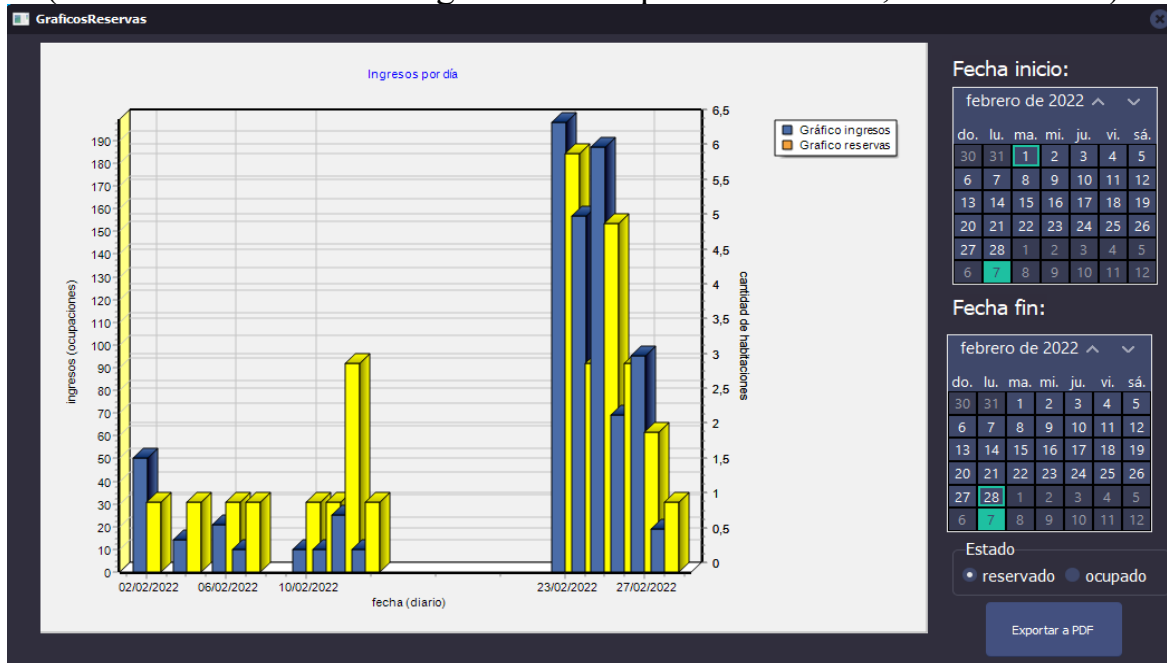
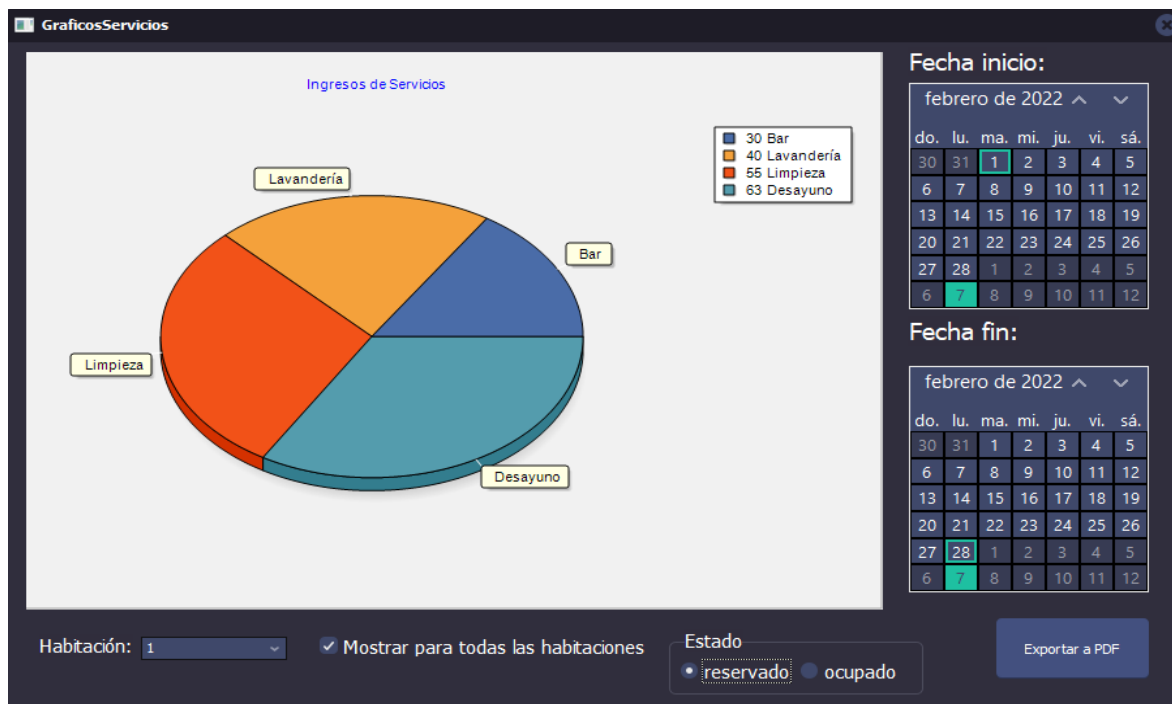


Gráfico de servicios: muestra los servicios reservados en febrero para todas las habitaciones.



3.7.4.- Especificación de la navegabilidad entre pantallas

Anteriormente hemos mostrado unos diagramas sencillos de la aplicación, pero en este punto mostraremos de manera más detallada como se navega entre las distintas pantallas. Dada la extensión de los diagramas, véanse las imágenes adjuntas de 'navegabilidad escritorio' y 'navegabilidad android'. En dichos diagramas, las flechas con dos líneas cruzadas simbolizan que los clientes no pueden acceder a esa pantalla (al menos, por esa ruta).

4.- Construcción/Compilación del sistema.

En este apartado, hablaremos sobre todo de la codificación o las técnicas de programación utilizadas en este proyecto.

4.1.- Acceso a datos

Uno de los apartados más importantes de la codificación en este proyecto es el acceso a datos, dado que la gran mayoría de acciones o funcionalidades requieren de una operación en base de datos. Ya hemos mencionado al principio de la necesidad de usar MyDAC en la versión android, que es el homólogo de FireDAC. En Delphi contamos con este tipo de componentes que nos permiten acceder a una base de datos de manera bastante cómoda, ya que las tablas de dicha base son representadas o referenciadas directamente como cualquier otro componente de un formulario (botón, label, etc). Dado que la extensión de nuestra base no es demasiado pequeña, contamos con un formulario a parte al que hemos llamado “Tablas”. No hemos mostrado su diseño dado que no es relevante, solo lo queremos para almacenar los componentes de acceso a datos.

Así pues, una vez hemos creado un elemento FTable o MyTable por cada tabla, podemos acceder a ellas y recorrerlas, realizar operaciones de consulta, insert, update, delete, etc. Para recorrer una tabla, comúnmente utilizaremos un bucle while, que no terminará hasta que la propiedad de la tabla 'eof' sea verdadera (indica si hemos llegado al final de la tabla). En cuanto a los campos de la tabla, al añadir un FTable, automáticamente se crean elementos que representan a sus campos, los cuales se designan con el nombre de la FTable seguido del nombre del campo. Podemos consultar o modificar el valor del campo accediendo a su propiedad 'value'.

Alternativamente, si necesitamos hacer consultas complejas, hacemos uso de una FDQuery, donde podemos introducir código SQL directamente, pero sólo para consultas. En este proyecto contamos con varias FDQuerys en la versión de escritorio, ya que son útiles para las consultas que necesitamos para realizar los informes, pero usualmente,

con tener una y cambiar su código SQL suele bastar.

Como indicamos al principio de este documento, la base de datos no será local en ningún caso, dado que queremos acceder desde distintos dispositivos. Debemos asegurarnos de que la base esté online, o de lo contrario la aplicación no funcionará al intentar realizar operaciones con la misma. En este documento se adjunta un script de SQL en caso de que se desee tener la BD en local aunque requiere que se cambie el servidor en el componente correspondiente.

4.2.- Creación de componentes en tiempo de ejecución

También hemos mencionado este punto en varios apartados anteriores. Delphi permite añadir cualquier tipo de componente de manera dinámica. Para guardar una referencia a cada componente no predefinido, crearemos un array para dichos componentes. Por ejemplo, para los paneles de las habitaciones, tendremos un “Array of Tpanel”, al que no le damos una longitud hasta saber la cantidad de habitaciones con las que contamos. Para ello simplemente accedemos a la propiedad RecordCount de la FDTTableHabitaciones. Una vez le pongamos la longitud necesaria, creamos un bucle for que recorra esa misma cantidad de veces, creando un panel por cada iteración. En cada vuelta, se creará un Tpanel nuevo, al que le pondremos una posición distinta. En el caso de las habitaciones, le damos valor a su propiedad Tag (el valor del nº de habitación) para así distinguir más tarde distinguir qué panel está llamando a otro procedimiento. Dicho de otra manera, si queremos que ese panel abra el formulario de reserva y queremos saber que ese panel representa la habitación 2, en su tag le habremos puesto un 2. De la misma manera podemos acceder a otras propiedades como OnClick o PopupMenu para que a los nuevos componentes se les pueda dar funcionalidad (en click izquierdo o derecho).

Esta técnica las aplicamos en distintas situaciones: en los paneles de las habitaciones y sus botones, en el panel de la pantalla mensual, en todos los checkboxes que representan a los servicios, incluso en la construcción del informe dinámico (de hecho, la pantalla de parámetros de dicho informe también se construye de la misma forma). Junto con la asignación de colores a los paneles, y teniendo en cuenta de que en algunos casos hay que hacer consultas a bases de datos, este tipo de prácticas puede provocar que el programa tarde un poco más en ejecutar, por lo que en cualquier caso, este tipo de código debemos ponerlos en funciones o procedimientos a parte y ejecutarlos sólo cuando sea necesario (por ejemplo, no re-creamos los paneles de habitaciones hasta que añadamos una nueva). Otro aspecto importante de estas partes del código es que si hay

que recrear algunos componentes, deberíamos borrar primero los que existieran antes, debido a que se pueden superponer. En la pantalla de mes, por ejemplo, donde la disposición de los días cambia dependiendo del mes, pueden salir días de otros meses a parte del mes en el que estamos, por lo que hay que vaciar el panel antes de volver a crearlos (añadimos dicho código en las funciones o procedimientos que hemos mencionado).

4.3.- Visibilidad y compartimiento de información entre formularios

Dado que los formularios o pantallas están entrelazados entre sí, deben poder compartir información o tener variables públicas que puedan ser visibles desde otras pantallas. Por ejemplo, si tenemos la necesidad de que al abrir el FormularioDiario, aparezca la habitación y el día del panel en el que hemos hecho click, tenemos dos opciones. En primer lugar, podemos tener una variable pública en el formulario de destino, en este caso, en FormularioDiario, por ejemplo `nhabitacion` de tipo `integer`. Si queremos que los paneles de habitación (o del mes) que hemos creado anteriormente le pasen el nº de habitación, en el método de dichos paneles se accederá a esta variable y se modificará. Por ejemplo: `FormularioDiario.nhabitacion := Panel.Tag` (recordemos que tenemos guardada la habitación en la propiedad `tag`). De este modo, al activarse el formulario de reserva, se sabe la habitación que se está tratando.

Alternativamente, si no queremos tener variables públicas, podemos tener métodos públicos. Es el caso del formulario `periodo`, que cuenta con las funciones “`modoReserva`” y “`modoAnular`”. Dado que tenemos dos botones para abrir dicho formulario, cada uno de ellos llama a la función correspondiente, que le indicará al formulario de `periodo` cuál va a ser su propósito, si reservar o anular.

Del mismo modo, si contamos con funciones o trozos de código que necesitamos repetir en distintos formularios, la mejor opción es situarlas en un formulario que sea visible para todos. En nuestro caso, el formulario “`Tablas`” casi siempre está importando en cada uno de las demás pantallas, dado que casi todas necesitan realizar operaciones de BD, por lo que en dicho formulario tendremos algunas funciones comunes, como por ejemplo, una función que rellena un combobox con todos los nº de habitaciones, u otra para hashear una contraseña o encriptar / desencriptar una cadena. Puntualizamos que para importar un formulario, hay que añadir el nombre de su unidad a la sección “`uses`” (`uses Unit3, Unit4; etc`).

En este apartado podríamos incluir otro punto, que es el de la reutilización de

formularios. No es un aspecto demasiado importante o recurrente en este proyecto, pero dada la cantidad de formularios que hemos creado, es conveniente ahorrar nuevas pantallas siempre que podamos. El caso más obvio es el del FormularioPeriodo que hemos mencionado hace poco. Según el modo que le indiquemos hace una cosa la opuesta. Podemos jugar con la visibilidad de los componentes, con cambiar el nombre de algunas etiquetas, etc. para dar la sensación de que el formulario es distinto. Otros casos de reutilización de formularios son todos los de alta de elementos (ya que permiten crear o editar) o el informe de factura, que se reutiliza como historial de cliente.

4.4.- Correos, encriptación, fechas y otros.

Incluiremos los siguientes aspectos o funcionalidades, algunos de ellos relacionados con seguridad, para que la aplicación se asemeje en cierta medida a un programa real y serio.

- Correos: A la hora de crear un usuario, pediremos un correo real para poder completar el alta de usuario. Para verificar que el correo indicado es el del usuario, mandaremos un email a esa dirección utilizando otra dirección que hemos creado para este proyecto. En primer lugar, lo que haremos es pasar el correo por una función de Regex (isMatch) para asegurarnos que tiene un formato válido. En tal caso, tenemos que construir o incluir 3 elementos (que tienen sus propios componentes en Delphi): un protocolo SMTP, un SSL handler socket, y un mensaje. En primer lugar, en el SSL debemos añadir los métodos de verificación, host, destino, etc que normalmente será gmail. En el SMTP incluiremos las credenciales de la cuenta del hotel, la que remitirá los emails. Le asignamos el handler anterior y por último, construimos el mensaje, que contendrá el destinatario, asunto, y el mensaje, donde incluimos el código de verificación, generado aleatoriamente. Con el método send del SMTP, podemos enviar el mensaje y el usuario debería recibirlo en su bandeja de entrada.

Nota importante: dado que estamos usando una cuenta particular para enviar esos correos, no podemos usarla tal cual para mandar correos en una aplicación no verificada. Hasta hace poco (30 de mayo, de hecho) se podía simplemente activar el acceso a aplicaciones poco seguras en gmail, pero Google está empezando a restringir esa opción por lo que debemos activar la verificación en dos pasos de la cuenta y crearle una contraseña de aplicación, la cual podremos poner en nuestro código para utilizar la cuenta.

- Encriptación y hash: Dicho lo anterior, no queremos poner una contraseña en texto plano dentro del código. Para ello utilizaremos unas funciones de encriptación y desencriptación (UTF8 encoding), a las cuales les pasamos una clave, cualquiera,

siempre y cuando usemos la misma en los dos sentidos. Por tanto guardaremos la contraseña de aplicación encriptada, y la hora de pasársela al SMTP, la mandaremos a la función de desencriptar.

Por otra parte, queremos guardar contraseñas de usuarios. Lo habitual es guardarlas como un hash, dado que no queremos que se puedan consultar en base de datos, sino que a la hora de que un usuario haga login, la contraseña introducida se convierta también en hash y se compare con la que hay en la BD. Para ello contaremos con otra función que creará un HashMessageDigest5 (MD5), con el cual haremos las operaciones pertinentes.

- Fechas: Una parte importante de este proyecto es manejar y operar con fechas. En muchas partes del código debemos recorrer un periodo de fechas, hacer una consulta en BD con fechas, o comparar fechas, etc. En la cláusula `uses` importaremos `System.DateUtils`, que nos permitirá realizar dichas operaciones. Por ejemplo, en la pantalla mensual, donde el calendario se genera a partir de cierta fecha, hacemos uso de varios métodos de ese import. Si abrimos el mes a 15 de febrero, podemos saber que tenemos 28 días con el método `DaysInAMonth`, podemos saber en qué día de la semana cae el día 1 con `DayOfTheWeek`, podemos obtener el día, mes o año de una fecha, o construir una fecha si tenemos esos 3 elementos con `EncodeDate`. Con toda esas herramientas hemos podido construir el calendario y realizar otras operaciones, como colorear el mismo calendario, recorriendo cada día para saber cuál está reservado y cual no.

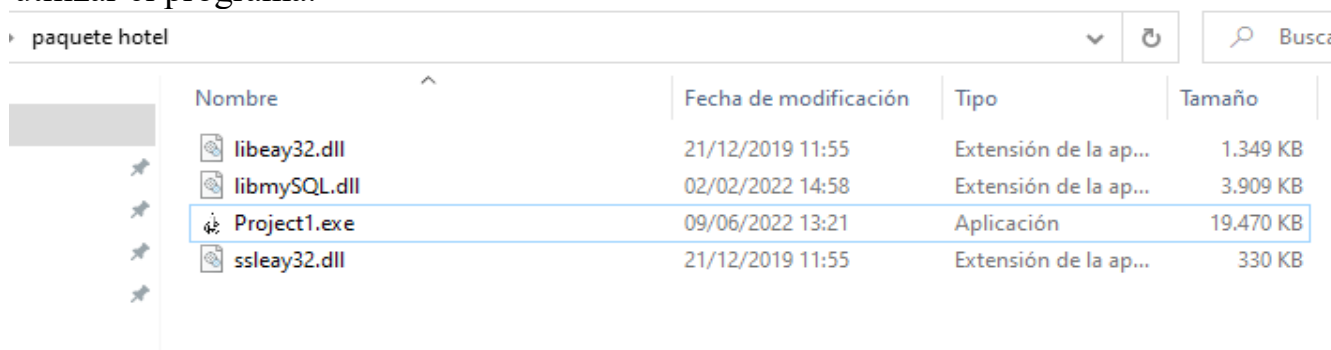
- Atajos de teclado: Por último, destacamos el uso de atajos de teclado para facilitar el uso de la aplicación. Para habilitar los atajos en cierto formulario, tenemos que poner su propiedad `KeyPreview` como `true`. A partir de ahí, podemos añadir los atajos que queramos en el método `FormKeyDown`, que recoge cada vez que se pulsa una tecla en dicho formulario. En la mayoría de formularios hemos habilitado la tecla de escape, por ejemplo, para cerrar esa pantalla y volver a la anterior. En las pantallas Principal y PantallaMes, se puede navegar entre fechas con las flechas izquierda y derecha (avanzando o retrocediendo un día o un mes). Por otra parte, el menú de la pantalla principal también puede manejarse con la tecla `Alt` junto con la letra subrayada de cada elemento del menú. Si queremos abrir el informe de Factura, podemos pulsar `Alt + I + F`. También hemos prestado atención, para mayor comodidad, a la propiedad `Tab Order` de los formularios, para indicar en orden en el que se salta de un componente a otro con el tabulador, ya que el orden inicial puede estar alterado.

En cuanto a la compilación, al tratarse de Delphi, no hay ninguna complicación. Al pulsar el botón de ejecutar se compila y se genera un ejecutable en la carpeta win32. Podemos mover ese ejecutable a otra carpeta, pero debemos tener en cuenta que las dll y otros archivos necesarios para el proyecto deben ir en la misma carpeta. En la versión android, tenemos dos carpetas, win32 y android, ya que como hemos dicho anteriormente, el proyecto también se puede compilar para ser ejecutado en el escritorio. En la carpeta de android se crean los archivos propios de un aplicación móvil, assets, resources, classes, etc.

5.- Manual de instalación y uso de la aplicación.

5.1.- Manual de instalación

Como acabamos de indicar, en la versión de escritorio contamos con un ejecutable, por lo que con tenerlo junto a los archivos necesarios, es suficiente para poder empezar a utilizar el programa.



Nombre	Fecha de modificación	Tipo	Tamaño
libeay32.dll	21/12/2019 11:55	Extensión de la ap...	1.349 KB
libmysql.dll	02/02/2022 14:58	Extensión de la ap...	3.909 KB
Project1.exe	09/06/2022 13:21	Aplicación	19.470 KB
ssleay32.dll	21/12/2019 11:55	Extensión de la ap...	330 KB

Podemos crearle un shortcut en el escritorio si lo deseamos, etc. En cuanto a la versión android, debemos ir al directorio donde tengamos el proyecto > Android > Debug > Nombre_del_proyecto > bin y allí encontraremos el apk. Lo pasamos al móvil con un usb o por google drive, por ejemplo, y lo instalamos. También puede instalarse desde el entorno de Delphi, por supuesto, si seleccionamos en los recuadros superiores “android 32 o 64 bit” y el dispositivo en cuestión. Recordamos que este último debe tener el modo depuración activado (en herramientas de desarrollador). Sin embargo, reiteramos que hemos usado los componentes de MyDAC para esa versión, y que si se tiene Delphi instalado, pero no tenemos dichos componentes, es probable que todos esos componentes se eliminen del proyecto, dejándolo inservible, por lo que para probar la app móvil, recomendamos el primer procedimiento en lugar de hacer esto último.

5.2.- Manual de uso de la aplicación (escritorio).

En este punto vamos a explicar cómo poner en práctica todas las funcionalidades de la versión de escritorio.

5.2.1.- Registrarse

Lo primero que necesitamos para poder utilizar la aplicación es una cuenta de usuario, por lo que el primer paso será registrarse.

Para registrarnos, en la pantalla de Login, pulsamos en 'Regístrate aquí'. Aparecerá un formulario modal en el que cumplimentar los datos del usuario. Es obligatorio rellenar todos los campos, y el correo debe tener un formato válido (y existir). Una vez rellenado todo, pulsamos 'Aceptar'. Si el correo existe, se le habrá enviado un correo con un código de verificación. Copiamos el código y lo pegamos en la caja modal en la que nos aparecerá.

Nota: Si el identificador de cliente que hemos introducido no existía en el sistema, se nos pedirá que lo creamos. Nos saldrá otro formulario para introducir su nombre y apellidos. Al pulsar aceptar, se dará de alta también el cliente. Si el ID de cliente sí existía, no se nos pedirá crear el cliente, sino que se le asignará el ya existente.

Si el código de verificación se introduce correctamente, veremos un mensaje del éxito de la operación y podremos utilizar la cuenta para entrar al sistema.

5.2.2.- Recuperar contraseña

Si hemos olvidado nuestra contraseña, podemos pulsar, en la pantalla de Login, la opción de 'Recuperar contraseña'. Nos saldrá una caja de diálogo en la que debemos introducir nuestro correo. Si es válido y existe, se le mandará un correo con un código. En tal caso aparecerá una nueva caja para introducir dicho código.

Si el código es correcto, aparecerán dos nuevas cajas para introducir la nueva contraseña dos veces. Si coinciden, la nueva contraseña se habrá guardado con éxito.

5.2.3.- Visualizar y navegar entre fechas

Si queremos consultar el estado de las habitaciones en distintas fechas, necesitaremos saber como navegar entre esas fechas. Recordamos que en la pantalla principal vemos

esos estados para todas las habitaciones en un día concreto, y en la pantalla mensual vemos el estado de una sola habitación durante un mes concreto.

En la pantalla principal tenemos hasta 5 controles para navegar entre fechas. Para saltar a un día concreto, utilizamos el calendario que hay a la derecha, o alternativamente, el DateEdit que muestra la fecha junto a “Fecha seleccionada”. Para volver al día actual, pulsamos “Volver al día de hoy”, a la izquierda. Para saltar un día hacia delante o hacia atrás, pulsamos los botones “Día anterior y día siguiente”, a la izquierda y derecha de la pantalla. Alternativamente, podemos usar las flechas izquierda y derecha para el mismo propósito.

Para visualizar la pantalla mensual, pulsamos en el botón “Abrir mes” de la habitación que queramos. En cualquier caso, si queremos visualizar el mes de una habitación distinta, no será necesario cerrar la pantalla y volver a abrir la de otra distinta. En PantallaMes, podemos usar el combobox (desplegable) para cambiar la habitación. De la misma manera que en principal, tenemos un DateEdit para saltar a fechas concretas, y dos botones para avanzar o retroceder un mes (las flechas también funcionan).

5.2.4.- Reservar o administrar un día concreto

Para abrir el formulario de administración de una habitación para un día concreto, tenemos dos opciones. En la pantalla principal, podemos hacer click derecho > administrar, o en la pantalla mensual, podemos hacer click en el día que queramos.

Esto abrirá el formulario diario, siempre que tengamos permiso. Si la habitación esta libre, podemos reservarla u ocuparla seleccionado una opción en los radio botones. En el cuadro de cliente debemos introducir un identificador de cliente (si somos administrador). Si no existe, se nos mostrará una pantalla modal para crearlo sobre la marcha. Pulsamos aceptar para guardar los cambios (la nueva reserva u ocupación). Si somos cliente, nuestro ID se pondrá automáticamente.

Para reservar servicios, debemos volver a abrir el formulario una vez se haya reservado u ocupado. Seleccionamos los checkboxs de los servicios que deseemos. Si somos administrador, se nos permite cambiar el precio final de la habitación. Pulsamos Aceptar para guardar los cambios.

Si queremos eliminar una reserva, basta con entrar al formulario, seleccionar el radio botón “libre” y pulsar aceptar. La reserva y servicios serán cancelados.

5.2.5.- Reservar o anular un periodo

En la pantalla principal contamos con dos botones, “Reservar periodo” y “Anular periodo”. Si queremos reservar un periodo, se nos abrirá un formulario donde indicaremos la fecha de inicio y fin (se incluyen ambas en el periodo) el cliente (si somos admin), la habitación y los servicios (marcamos los que queramos, luego se pueden cambiar en cada día concreto).

Si existe alguna reserva o ocupación en el periodo indicado, no se llevará a cabo la operación. Recomendamos visualizar el estado mensual de la habitación antes de hacer una reserva de un periodo para la misma.

Para anular un periodo, con el otro botón, abriremos el mismo formulario en el que solo nos pedirán una fecha de inicio y de fin y la habitación. Si somos admin, se eliminan todas las reservas del periodo. Si somos cliente, solo se borran las que estén a nuestro nombre.

5.2.6.- Visualizar todas mis reservas / ocupaciones

Como cliente, si queremos tener un informe de todas nuestras reservas realizadas en el hotel podemos elegir las opciones de menú Informes (Factura o historial). En Factura, debemos introducir las fechas de inicio y fin, la habitación y si es reserva u ocupación para obtener un informe acotado a esos parámetros. Con la opción de historial, obtendremos todo nuestro historial, reservas y ocupaciones, en cualquier habitación y fecha.

Como administrador tenemos las mismas opciones, pero Factura incluirá cualquier cliente, e historial requerirá que introduzcamos un ID de cliente.

5.2.7.- Visualizar itinerario de servicios y otros informes y gráficos

Otras opciones sólo para administradores.

Para ver los servicios que se deben prestar en un día concreto, en la pantalla principal debemos navegar al día que queramos y pulsar Informes > Itinerario de servicios. Se nos mostrará un informe con las habitaciones que han contratado servicios, agrupadas por nombre de servicio.

Para imprimir un listado de cualquier entidad de la base de datos, podemos ir a Informes > Informe Dinámico. Se nos mostrará una pantalla en la que debemos escoger, en primer lugar, la tabla de la que queremos mostrar información, de la columna izquierda. En segundo lugar, sus campos se mostrarán en la columna central. Por defecto estarán todos seleccionados, podemos deseleccionar los que no nos interesen. Por último, escogemos en la columna derecha el campo por el que queremos ordenar, y si es de manera ascendente o descendente.

Pulsamos generar y obtendremos el informe con los parámetros especificados. Para imprimir o guardar cualquier informe, en la pantalla de previsualización del mismo, tenemos las opciones de print report y save report.

En cuanto a los gráficos, podemos seleccionar 2 de los disponibles en el menú de Gráficos. Si queremos visualizar un gráfico de las reservas, escogemos la opción de Ingresos – reservas, que mostrará un gráfico de barras. Podemos escoger el periodo y cambiar entre reservas y ocupaciones con los controles de la derecha. Podemos pulsar en “Exportar a PDF” para obtener una instantánea del gráfico en pdf.

Si queremos un gráfico de los servicios contratados, seleccionamos Ingresos – servicios, que mostrará un gráfico de sectores en proporción a los ingresos de los servicios. Nuevamente podemos escoger el periodo, reserva u ocupación, y la habitación (o todas las habitaciones) con los controles disponibles, y exportar a pdf si lo deseamos.

5.2.8.- Crear habitaciones, servicios, etc.

Como administrador, podemos crear habitaciones, servicios, temporadas y usuarios desde el menú (Crear). Los clientes solo se pueden crear al dar de alta un usuario o una reserva. Para cualquier opción, se abrirá un formulario en el que debemos rellenar todos los campos y pulsar “Crear”. Las habitaciones también se pueden crear en la pantalla principal, haciendo click derecho sobre el panel central de las habitaciones.

Otro requisito para crear elementos es no repetir la información de la clave primaria. Es decir, una habitación nueva debe tener un número nuevo. Lo mismo ocurre con el identificador del cliente, el nombre del servicio, y el correo y nombre de usuario de un usuario. Las temporadas son una excepción dado que nos fijamos en si los periodos se solapan con otros existentes.

5.2.9.- Editar habitaciones, servicios, etc.

El procedimiento es similar al punto anterior, pero en este caso, al pulsar Editar + cualquier opción, se nos pedirá que introduzcamos la clave principal del elemento a modificar (nº de habitación, nombre de servicio, id de cliente). Si dicha clave existe, se abrirá el formulario correspondiente con la información existente. Podemos modificarla y pulsar Editar para guardar los cambios.

Con las temporadas hemos hecho una excepción, de nuevo, dado que manejar la edición de las nuevas fechas puede ser complicado, por lo que en su lugar, tenemos la opción de “Borrar temporada”. Simplemente introducimos una fecha y aquella temporada que contenga esa fecha será eliminada, por lo que el periodo quedará libre para que se puedan crear nuevas temporadas en él.

5.2.10.- Preguntas frecuentes

¿Puedo crear un nuevo usuario con un cliente que ya existía?

No, solo puede haber un usuario asociado a un cliente. Si no recuerda la contraseña, puede recuperarla con la opción de “Recuperar contraseña”.

¿Cuándo debo marcar como una habitación como ocupada?

Será tarea de un administrador, del recepcionista, por ejemplo. Solo se puede marcar como ocupada una habitación en el día de hoy o con 7 días de retraso. Si la habitación está reservada y se pasa el día, se puede ocupar o liberar, pero lo ideal es que en el día a día, el administrador se encargue de confirmar las reservas o borrarlas si se cancelan.

Me he equivocado al reservar una habitación o servicio, ¿puedo corregirlo?

Como cliente, sí, siempre y cuando sea referente a un día futuro (no se ha consumado esa reserva). Se puede acceder al formulario de administración diario para quitar los servicios que no se deseen, o anular la reserva. También se puede anular un periodo entero, y por consiguiente, todos los servicios.

Si modifico el precio de un servicio o habitación, ¿Se modifica el precio de las reservas ya realizadas?

No, el precio de una reserva u ocupación (que incluye los servicios contratados) se guarda a parte, y no se recalcula a partir de los precios actuales de habitación y servicio. Si queremos que el nuevo precio aplique a una reserva, debemos liberarla y volverla a reservar (o cambiar el precio manualmente como admin). La misma lógica aplica si

borramos una temporada.

¿Como puedo distinguir mis reservas de las de otros clientes?

Si por ejemplo sabemos que hemos reservado ciertos días de un mes, pero hay mas reservas de otros clientes en el mismo mes y nos causa confusión, recomendamos usar la opción de menú Informes > Factura e indicar el mes o periodo, y la habitación en cuestión. De esa manera podemos ver rápidamente qué días son los que menos reservado nosotros.

5.3.- Manual de uso de la aplicación (android).

En este punto explicaremos como utilizar las funcionalidades de la app móvil.

5.3.1.- Registrarse

Al lanzar la app, de la misma manera que en escritorio, obtendremos la pantalla de Login. En ella podemos introducir los datos, todos son obligatorios excepto el nombre apellidos, en caso de que el ID de cliente ya exista. Si pulsamos aceptar y no existe, se nos requerirá que los cumplimentemos. Las mismas reglas aplican para el correo, nombre de usuario y contraseña. En este caso hemos prescindido de los códigos de verificación, más adelante explicaremos por qué.

5.3.2.- Visualizar y navegar entre fechas

Una vez entremos con nuestras credenciales, en la parte superior de la pantalla principal, bajo la barra del título, tenemos dos botones para avanzar y retroceder un día, y un DatePicker que abrirá un calendario para seleccionar fecha concretas.

De la misma manera, si pulsamos “Abrir Mes” obtendremos la pantalla mensual para la habitación en concreto. Podemos usar, de la misma forma, los botones para cambiar al siguiente o anterior mes, y el datepicker para saltar a una fecha concreta. También contamos con un combobox de habitaciones para cambiar de habitación sin tener que volver a la pantalla anterior.

5.3.3.- Reservar o administrar un día concreto

Para este propósito de nuevo tenemos dos formas de acceder. Desde principal, pulsamos en el cuadro de la habitación que queramos administrar, o bien, desde la pantalla mensual, pulsamos sobre el cuadro del día que deseemos. Esto abrirá un formulario de administración similar al de la versión escritorio.

Seleccionamos el nuevo estado con los radio botones, y los servicios con los checkboxes. Recordamos que los servicios no se mostrarán hasta que el estado de la habitación sea distinto de libre, es decir, hay que reservar/ocupar, aceptar, y luego volver a entrar para modificar los servicios y/o precio. El ID de cliente también se debe especificar, si somos un administrador. En caso de que el cliente no exista, debemos crearlo previamente, no se nos permite la opción de crearlo sobre la marcha.

5.3.4.- Reservar o anular un periodo

Estas dos opciones se encuentran en el menú de la pantalla principal. Si tocamos el icono superior izquierdo, se mostrará el menú lateral. Allí, pulsamos “Reservar” o “Anular” para acceder a estas opciones.

En la reserva de periodo deberemos seleccionar las fechas de fin e inicio, el cliente (si somos admin, recordamos que el cliente debe existir previamente) y la habitación, y los servicios, opcionalmente. Si ya hay reservas en el periodo especificado para la habitación, no se llevará a cabo la operación.

Si queremos anular un periodo, indicamos las fechas de inicio y fin y la habitación. Al pulsar aceptar se anularán todas las reservas de dicho periodo. Si somos cliente, solo se anulan las nuestras.

5.3.5.- Crear habitaciones, servicios, etc.

De nuevo, son opciones de administrador. En este caso, incluimos la opción de crear clientes, dado que no podemos crearlos a la vez que reservamos. Todas estas opciones se encuentran en el menú lateral de la pantalla principal. Seleccionamos la que deseemos y se abrirá un formulario para cumplimentar la información del elemento que queramos crear.

De la misma manera, no podemos repetir información clave, como el nº de habitación, ID de cliente, etc. pulsamos aceptar para crear el elemento y automáticamente volveremos a la pantalla principal.

6. - Conclusiones

Este proyecto de desarrollo puede ser útil para pequeñas empresas o grupos de trabajo que deseen llevar un control de reservas, en este caso, de habitaciones de hotel, aunque con los cambios oportunos, se podría transformar en un programa de reservas de otro tipo de cosas, como aulas de instituto, por ejemplo. Uno de los principales objetivos era darle cierto dinamismo y escalabilidad al proyecto, dado que una empresa, en el mundo real, puede cambiar y crecer, por lo que una interfaz con elementos fijos puede ser contraproducente si queremos ampliar sus límites.

Otro aspecto importante que hemos tenido en cuenta es la presentación de la información en pantalla. Es importante transmitir la información justa y necesaria y de manera cómoda a la vista, para lo cual hemos utilizado sobre todo los paneles y los colores. También hemos incluido controles intuitivos para navegar entre fechas y pantallas, por lo que consideramos que cualquier usuario puede aprender fácilmente a manejarse con la aplicación.

Otro objetivo para este proyecto era la inclusión de distintos perfiles, dado que en principio, la aplicación estaba pensada para ser utilizada por un único administrador. Al hacer uso de una base de datos remota e incluir el código necesario para diferenciar y controlar las posibilidades de clientes y administradores, hemos logrado una aplicación con la que varias personas puedan trabajar o utilizar a la vez e incluso ver los cambios de información prácticamente en tiempo real.

Donde más problemas hemos tenido, sin duda, ha sido con la versión móvil, y prueba de ello es que algunos aspectos de la versión escritorio no han sido incluidos. Esto se debe a que en la mayoría de aspectos, la codificación y construcción de la app es similar, pero hay ciertos puntos que no se permiten realizar de la misma manera. Algunas funcionalidades, la más crucial de hecho, que la de acceso a datos, la hemos conseguido solventar con un componente de terceros, pero otras, como el uso de correos para mandar códigos de verificación, no han sido posibles en última instancia, dado que no podemos incluir los archivos dll necesarios en el apk. La única alternativa en este caso concreto era realizar lo que se conoce en android como un “intent” pero esto simplemente nos abre la opción de mandar un correo, utilizando el que tengamos en el móvil, y no el que hemos creado para el proyecto, por lo que simplemente decidimos prescindir de esa característica en android.

Otro problema que surge con el correo es el “acceso a aplicaciones poco seguras”. Lo

normal sería tener un servidor de correos propio, pero dado que en este caso simplemente queremos tener una cuenta gmail para mandar correos, Google nos pone bastante escollos a la hora de usar esa opción, por seguridad. Finalmente hemos optado por la opción de crear contraseña de aplicación, que es una opción algo más segura que simplemente usar la contraseña de la cuenta en el código (aunque la hayamos encriptado).

Otro aspecto de android bastante problemático ha sido el concepto o uso de pantallas o cajas “modales”. En la versión escritorio, el alta de cliente está diseñada para realizarse en mitad de otras operaciones cuando fuese necesario (al reservar o crear un usuario). Esto se traduce en que una pantalla modal aparece para despachar rápidamente esa necesidad de crear un cliente, interrumpiendo momentáneamente el proceso que estábamos llevando a cabo. Pero en android, esta técnica o forma de codificar no ha sido posible. Por ejemplo en el caso de la reserva, al mostrar la pantalla de nuevo cliente, no se interrumpía el resto del proceso, por lo que aunque creáramos el cliente, el resultado era un mensaje de que “el cliente no existe”. Por tanto la solución ha sido requerir que el cliente ya exista de antemano. Es una opción menos cómoda, pero es lo mejor que se nos ha permitido implementar.

Desde otro punto de vista, es la primera vez que realizamos una aplicación android con Delphi, por lo que a parte de desarrollo, también ha habido una buena parte de investigación para saber como poner en marcha la app, solventar el tema de acceso a datos, etc. Por lo que consideramos que la envergadura, cantidad de opciones y extensión de la app móvil es bastante satisfactoria de por sí, aunque comparativamente sea menor que la versión de escritorio.

En conclusión, hemos cumplido con los objetivos propuestos para este proyecto, obteniendo un resultado que se asemeja a lo que esperábamos en un principio. Las aplicaciones cumplen con las funcionalidades y aspectos que queríamos implementar, los cuales hemos mostrado y explicado a lo largo de este documento.

7. - Bibliografía

- Documentación de Delphi: <https://docwiki.embarcadero.com/>
- Ayuda para Delphi: <http://www.delphibasics.co.uk/>
- Documentación MySQL: <https://dev.mysql.com/doc/>
- MyDAC: <https://www.devart.com/mydac/>
- QuickReport: <https://www.quickreport.co.uk/>
- Tutorial email en Delphi: <https://www.emailarchitect.net/easendmail/kb/delphi.aspx?cat=2>
- Archivos DLL para SSL: <https://indy.fulgan.com/SSL/?C=N;O=D>
- Guía MD5 Delphi: <https://delphidesdecero.com/tutoriales/calcular-md5-desde-delphi/>
- Uso de “modal” en Delphi / Android:
<https://docwiki.embarcadero.com/Libraries/Sydney/en/FMX.Forms.TCommonCustomForm.ShowModal>