HOTEL MANAGER

2º DAW

Marzo

CPIFP Los Enlaces

Mario Pérez Santolaria
Tutor: Sergio Pacheco

DOCUMENTACIÓN HOTEL MANAGER

Índice de contenido

1. Propuesta de proyecto: Hotel Manager

1.1. Título del proyecto

Aplicación web para la gestión interna hotelera de un hotel.

1.2. Nombre de la aplicación

Hotel Manager

1.3. Contexto del proyecto

Esta aplicación va dirigida a un negocio particular (un hotel en este caso) y relativamente pequeño, por lo que se ajusta a sus necesidades. El hotel necesita registrar clientes, las reservas que estos le pidan, llevar control de las facturas y servicios que ofrece.

1.4. Objetivo breve

Gestionar, registrar y llevar el control de la actividad del hotel en los aspectos más importantes.

DOCUMENTO DESCRIPCION DEL PROYECTO

2. Documentación de descripción del proyecto

2.1. Contexto del proyecto

2.1.1. Ámbito y entorno

El proyecto consiste en la realización de una aplicación web de gestión interna para un hotel. El hotel no pertenece a ninguna cadena, sino que está gestionado por una supuesta familia. Aunque no es un gran complejo, sí que cuenta con varios servicios que le convierten en un alojamiento que tiene su cabida en el mapa.

La familia gestiona el hotel y sus empleados utilizando métodos tradicionales, sin utilizar ningún software especializado.

El hotel ha pedido una aplicación web capaz de gestionar y llevar a cabo las reservas de sus clientes y los servicios que pueden solicitar, la gestión de las habitaciones, así como automatizar ciertas tareas con los servicios de mantenimiento y limpieza.

2.1.2. Análisis de la realidad

Actualmente el hotel no cuenta con ningún software especializado que les ayude a realizar su gestión, aunque ahora se utiliza alguna herramienta ofimática como hojas “Excel” o el “Word”. Por ello los clientes requieren una aplicación web desde cero que les automatice todas las tareas que ahora tienen que realizar con mucho trabajo y tiempo con las pobres herramientas ofimáticas para la gestión y automatizar también el contacto con sus servicios de mantenimiento y limpieza puesto que ahora se usa el teléfono o el boca a boca.

Existen en el mercado soluciones software para la gestión hotelera, pero son aplicaciones demasiado grandes y complejas que además de ser difíciles de manejar suponen un alto coste en licencias. Además, estas soluciones son genéricas y las modificaciones particulares que requieren nuestros clientes tienen un sobrecoste punitivo.

Por otro lado, nuestra aplicación aparte de ser sencilla de manejar está completamente hecha a medida de las necesidades y requerimientos del cliente por lo que éste solo pagará por lo que necesite.

2.1.3. Solución y justificación de la solución propuesta

Se han mantenido varias reuniones preliminares con los clientes para saber exactamente cuáles eran sus necesidades y básicamente eran tres: ahorro de tiempo en la gestión, una organización más eficaz, rápida y sin errores y una comunicación automatizada con cierto personal. Además, quieren una aplicación para uso interno, en ningún caso un portal web.

Por todo ello, se ha propuesto una aplicación web que permita una sola implementación (con el ahorro de coste que conlleva) que pueda usarse desde cualquier dispositivo dentro de la red del hotel. De esta forma la aplicación estará alojada en un servidor particular dentro del hotel y se podrá acceder a ella desde los distintos puestos o mostradores sin requerir de ninguna instalación. Además, esto permitirá interconectar la aplicación con distintos miembros del personal de limpieza y mantenimiento a través de tablets o teléfonos móviles y todo ello con una sola implementación. De esta forma incluso si algún equipo se estropea puede ser sustituido por cualquier otro dispositivo ya sea un pc, un portátil o una tablet sin necesitar de ninguna ayuda por parte de ningún personal especializado (de nuevo ahorrando costes).

Por último, nuestra solución no tiene ninguna clase de licencia de uso dado que es un

producto totalmente a medida del hotel. De esta forma el cliente se ahorra los costes de licencias y solo paga por el trabajo realizado. A la entrega, el código estará totalmente a disposición del cliente.

2.1.4. Destinatarios del proyecto

La aplicación podrá ser usada por diferentes perfiles de usuarios: Un perfil de administrador que tendrá acceso a toda la aplicación, incluida la configuración. Un perfil de mostrador que podrá realizar únicamente las tareas de mostrador. Y por último dos perfiles destinados a su uso en dispositivos móviles para los personales de limpieza y mantenimiento, cada uno adaptado a sus requerimientos.

2.2. Objetivo del proyecto

El objetivo del proyecto es realizar una aplicación web capaz de gestionar las reservas y estancias de los clientes del hotel, así como mantener el control en todo momento del estado de las instalaciones y mantenimiento del hotel.

Los administradores accederán a un panel de control donde tendrán la opción de registrar nuevos clientes, asignarles habitaciones, prepararles la factura de la estancia o reservarles alguno de los servicios del hotel.

También se encargará de poner horarios para el personal de limpieza y de comunicar incidencias al personal de mantenimiento.

Se pretende con este proyecto ayudar a este negocio en particular, proponiendo una organización clara, ágil y eficaz mediante un buen producto que sea perfectamente funcional y que haga que el hotel siga creciendo hasta ponerse a la altura de sus competidores.

2.3. The project's objetive

The aim of the project is to create a web application capable of managing the bookings and stays of the hotel customers, as well as to maintain control at all times of the state of the hotel's facilities and maintenance.

Administrators will access a control panel where they will have the option of registering new clients, assigning them rooms, preparing their stay invoice or booking any of the hotel services for them.

It will also be in charge of setting schedules for the cleaning staff and reporting incidents to the maintenance staff.

It is intended with this project to help to this bussiness in particular, proposing a clear, agile and efficient organization, through a good product which is perfectly functional and which makes the hotel keep growing until it catches up with its competitors.

2.4. Marco legal

La aplicación web está sujeta a la ley de protección de datos por el hecho de que debe recoger los datos personales de los clientes y tratarlos, siempre confidencialmente.

La aplicación no tendrá licencia de uso, el código estará a disposición del cliente en el momento de la entrega y podrá hacer con él lo que quiera.

DOCUMENTO DE ACUERDO DEL PROYECTO

3. Documento de acuerdo del proyecto

3.1. Requisitos

A continuación, se detallan los requisitos planeados para este proyecto.

Estos requisitos se han establecido en las distintas reuniones con el cliente. De esta forma se ha podido especificar lo que realmente necesita el cliente. Esto permite saber cuáles son los gustos del cliente y se puede priorizar en ciertos requisitos a la hora de planificar las tareas.

Los requisitos se han dividido en funcionales y no funcionales.

3.1.1. Requisitos funcionales

Los requisitos funcionales son aquellos que definen las distintas tareas que tiene que hace la aplicación.

Para esta aplicación se han pedido las siguientes:

ID	Requisito	Descripción
RF01	Gestión de clientes	La aplicación debe permitir gestionar la información de los clientes: consulta, inserción, edición y borrado.
RF02	Gestión de habitaciones	La aplicación debe permitir gestionar la información de las habitaciones: introducir las habitaciones en el sistema, editarlas y borrarlas.
RF03	Gestión de reservas	La aplicación debe permitir gestionar las reservas de las habitaciones: consultar el estado de las habitaciones, reservar y liberar habitaciones.
RF04	Facturas de estancias	La aplicación debe permitir la gestión de las facturas: consulta, emisión, edición e impresión en pdf.
RF05	Gestión de servicios y espacios	La aplicación debe permitir gestionar los servicios y espacios del hotel (gimnasio, salones, sauna, etc.): consulta, inserción, edición y borrado.
RF06	Reserva de servicios y espacios	La aplicación debe permitir gestionar las reservas de los servicios y espacios.
RF07	Gestión de plantilla de personal	La aplicación debe permitir la gestión de la plantilla del personal de limpieza y mantenimiento: consulta, inserción, edición y borrado.
RF08	Mantenimiento	La aplicación debe permitir la emisión de incidencias que recibirá el personal de mantenimiento, asignar las incidencias al personal, consultar el histórico de incidencias, marcar incidencias como resueltas, escalado de incidencias (una incidencia que no puede ser resuelta por el personal).

RF09	Limpieza	La aplicación debe permitir que el personal de limpieza pueda ver en tiempo real el estado de las habitaciones por limpiar en caso de check-out para que esté la habitación disponible lo antes posible.
RF10	Gestión de usuarios	La aplicación debe permitir al usuario administrador la gestión de usuarios (consulta, inserción, edición y borrado) y la asignación de perfiles.

3.1.2 Requisitos no funcionales

Los requisitos no funcionales son aquellos que definen las características que tiene la aplicación.

Para este proyecto se decide que sean las siguientes:

ID	Requisito	Descripción
RFN01	Backend	Lenguaje PHP y base de datos MySQL unidos por el framework Laravel.
RFN02	Frontend	HTML5, CSS3 con JavaScript, JQuery y librerías basadas en ambas.
RFN03	Accesibilidad	El programa podrá ser utilizado desde cualquier navegador moderno (siempre desde sus últimas versiones) en cualquier dispositivo.
RFN04	Usabilidad	Interfaz responsiva gracias a Bootstrap 5 para que se ajuste a cualquier pantalla. La aplicación tendrá dos formatos: Desktop para la gestión en general y móvil para el uso del personal de limpieza y mantenimiento.
RFN05	Aprendizaje	Uso sencillo con menús y botones con ayuda textual y con iconos para que cualquier operación posible sea absolutamente clara y evidente para el usuario.
RFN06	Seguridad de la red	La aplicación funcionará en red local evitando exponerse a internet.
RFN07	Seguridad dentro de la aplicación	Las distintas páginas tendrán los permisos de uso y acceso adecuados para los usuarios de la aplicación conforme a cuatro perfiles predeterminados: administrador, mostrador, mantenimiento y limpieza.
RFN08	Usuarios de la aplicación	Los usuarios tendrán un perfil de permisos y un usuario y una contraseña definida por el administrador.

RFN09	Concurrencia	Podrá haber tantos usuarios a la vez logueados en el sistema como se quiera.
RFN10	Rendimiento	El sistema tendrá un rendimiento fluido y sin bloqueos debido a la utilización poco intensiva tanto de CPU como de acceso a la base de datos debido a que todas las operaciones no requieren de grandes recursos.

3.2. Tareas

El análisis que se ha realizado sobre las tareas a realizar y su duración es el siguiente:

Tarea	Descripción	Nº Horas
T1	Investigación del mercado actual y posible solución. Objetivos del proyecto.	2
T2	Descripción del proyecto	5
T3	Documento de acuerdo del proyecto	15
T3.1	Requisitos	4
T3.2	Tareas a realizar	3
T3.3	Metodología y elección de un modelo de ciclo de vida	1
T3.4	Planificación temporal	4
T3.5	Presupuesto (gastos, ingresos y beneficio)	2
T3.6	Análisis de posibles riesgos y errores que puedan suceder. Creación análisis DAFO	1
T4	Contrato y condiciones de ambas partes	1
T5	Primer Sprint	79
T5.1	Análisis y diseño del primer sprint.	8
T5.2	Diseño del sistema y arquitectura de software: realización de diagramas.	20
T5.3	Realización de mockups.	10
T5.4	Reunión con el cliente.	3
T5.5	Desarrollo del core del sistema: montar bbdd, plantilla HTML y funcionamiento básico del backend.	4
T5.6	Desarrollo de frontend según los mockups e integración dentro del core del sistema	30
T5.7	Pruebas de software del primer sprint	4
T6	Segundo Sprint	55

T6.1	Reunión con el cliente.	3
T6.2	Análisis y diseño del segundo sprint.	4
T6.3	Cambios en el software del sprint 1	12
T6.4	Implementación del backend del proyecto	24
T6.5	Interconexión backend - frontend	8
T6.6	Pruebas de software del segundo sprint	4
T7	Implementación final: últimos retoques	6
T8	Pruebas completas de la aplicación web	8
T9	Documentación del proyecto	12
T10	Entrega al cliente	
-		Total: 183 h.

3.3. Metodología a seguir

La metodología que he escogido para este proyecto es una metodología ágil con modelo de ciclo de vida en espiral. En este modelo se escogen sprints y se incrustan en la planificación como una tarea adicional que implica revisar todo lo hecho hasta ese momento para después entregarlo al cliente, incrementándolo con los siguientes pasos. Se ha establecido un único sprint que concede al cliente la posibilidad de revisar el diseño del proyecto antes de empezar a codificar con el objetivo de que el cliente de el visto bueno, asegurándose así que el proyecto cumple con las exigencias definidas por el cliente.

3.4. Planificación temporal de las tareas

El Diagrama de Gantt ilustra la planificación de tareas que he tomado, y su duración provisional. Representa en forma de calendario el tiempo estimado que me supone hacer cada tarea y las fechas en las que se realiza. A continuación, muestro el diagrama de Gantt de este proyecto.

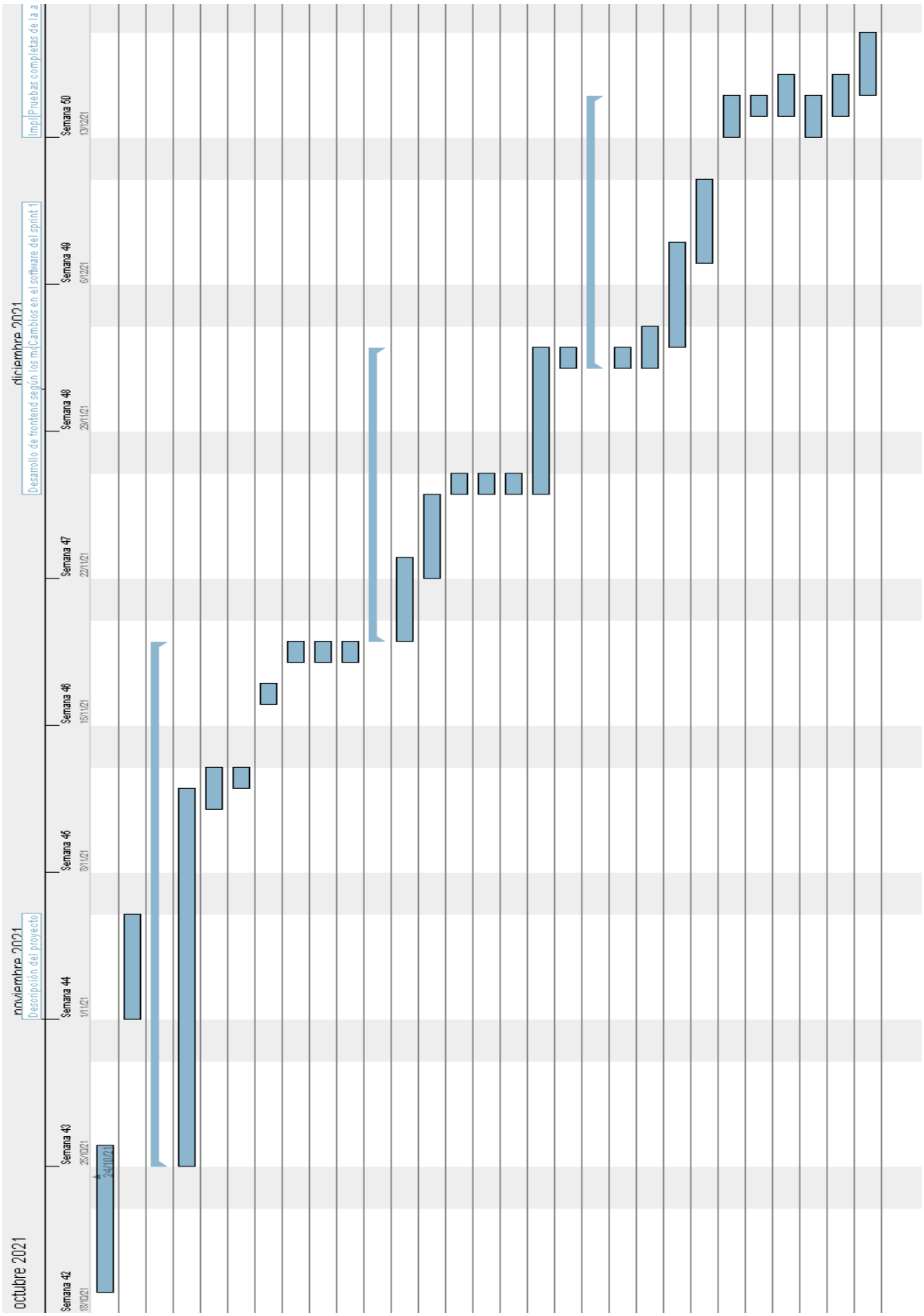


Figura 3.1 Diagrama de Gantt. Las fechas son estimadas.

3.5. Presupuesto

Las siguientes tablas reflejan un resumen de lo que es el presupuesto, que se puede encontrar completo en el anexo 8.1

3.5.1. Costes fijos y variables

A continuación, se muestran los costes fijos y variables. Los precios, salvo que se indique otra cosa, son por mes.

Costes fijos			
Partida	Meses	Precio/mes	Subtotal
Materiales	12	35 €	420 €
Licencia	3	18 €	54 €
Servicios (luz…)	3	170 €	510 €

Costes variables			
Partida	Horas	Precio/hora	Subtotal
Alquiler sala reuniones	6	11 €	66 €
Roles	183	85 €	15.555 €

Los materiales son cualquier elemento físico que se necesite utilizar en el proyecto, incluido el equipo. Con la licencia nos referimos al software que se necesita para programar (IDE). Finalmente, los servicios como la luz o la gestoría también se incluyen.

En cuanto a costes variables a contar el coste del alquiler de la sala donde tienen lugar las reuniones y los acuerdos, y los roles, que son cada fase del proyecto como trabajador (por ejemplo, analista).

3.5.2. Precio técnico de venta

Costes fijos	Costes variables	Coste total
774 €	15.621 €	16.395 €

Especificados los costes ahora tenemos que sumarlos, lo que nos dará el total de cada tipo de coste.

3.5.3. Beneficio total por desarrollo del proyecto

He decidido cobrar al cliente el coste total que supone el proyecto más un 3% de comisión para generar beneficios.

3.5.4. Presupuesto completo

El presupuesto completo se puede consultar en el anexo 8.1

3.6. Análisis de riesgos

Para considerar y anticipar los posibles riesgos que puede ocasionar la aplicación web, se ha hecho un análisis del programa con sus fortalezas y debilidades. El resultado se expone en la siguiente tabla (DAFO).

Debilidades	Fortalezas
Depender de un sistema únicamente	Mayor control sobre las tareas de mantenimiento

Que haya partes de la gestión que no estén contempladas por el programa	Mayor control sobre el estado de las habitaciones, así como la estancia del cliente Avisos al personal sobre el estatus de cada habitación en tiempo real Ahorro de tiempo Más corrección Mejor comunicación Todo bien archivado Ahorro económico Gracias a la generación automática de facturas, es más fácil llevar las cuentas
Amenazas	Oportunidades
El gran número de aplicaciones de gestión de hoteles	La posibilidad de comunicarse con el servicio de mantenimiento en tiempo real

3.7. Contrato / Pliego de condiciones

El contrato junto a las condiciones se adjunta en el anexo 8.2

DOCUMENTO DE ANÁLISIS Y DISEÑO

4. Documento de análisis y diseño

4.1. Modelado de datos. Análisis y diseño de la tabla de datos

A continuación, presento el diagrama de E/R sobre el que está construida la base de datos, y su diagrama relacional.

4.1.1. Diagrama E/R

El diagrama Entidad-Relación es una representación gráfica del contenido de una base de datos. Es la primera de las tres fases que existen para diseñar y organizar los datos estructurados.

Como se puede apreciar, existe una agregación, que engloba la relación clientes-reservas-habitaciones, de la que salen dos relaciones, como son generan y reservan.

El diagrama E/R de este proyecto es el que sigue:

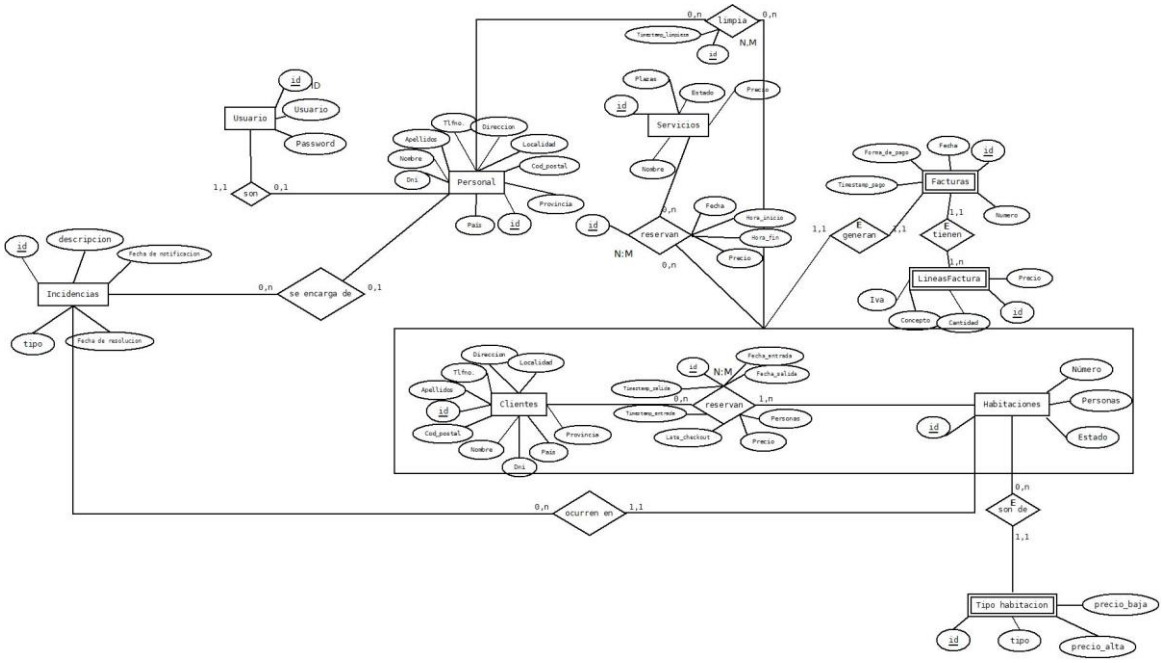


Fig. 4.1 Diagrama E/R

No obstante, el proyecto que tengo en mis manos ha ido cogiendo forma desde el comienzo, y es lógico que sufra cambios imprevistos. Uno de ellos es el del esquema Entidad-Relación de la base de datos. Se ha visto alterada una entidad, lo cual me lleva a rectificar el esquema y añadir los dos atributos que faltan:

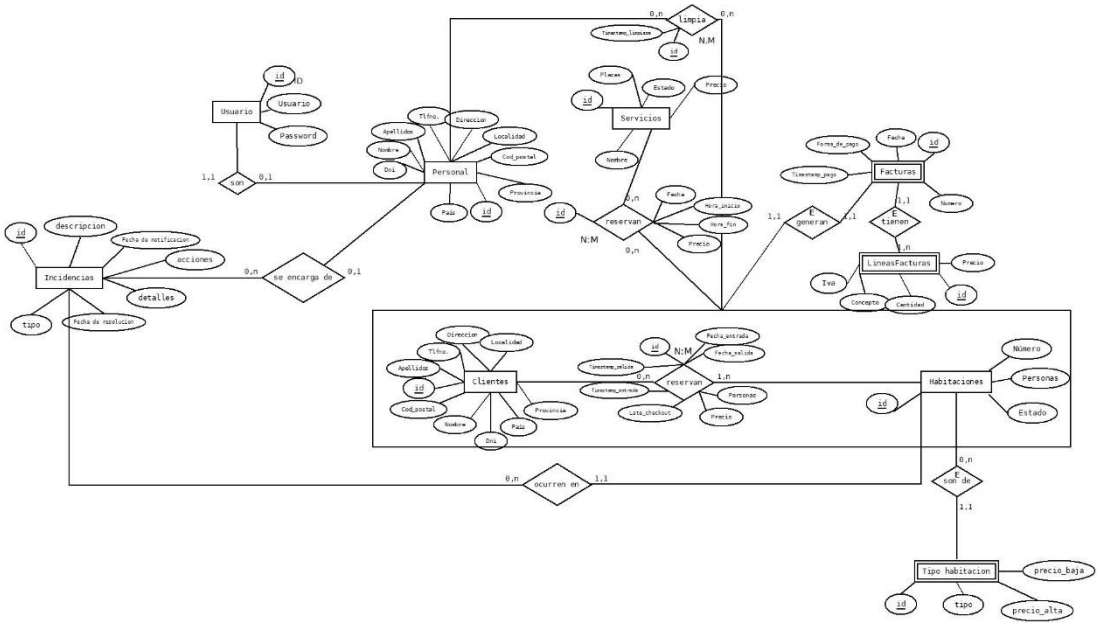


Fig. 4.2 Diagrama E/R modificado previsto

En él se observa como incluyo dos atributos más en la tabla Incidencias: detalles y acciones.

4.1.2. Diagrama relacional

La siguiente figura refleja el modelo relacional del diagrama E/R anterior.

CLIENTES (**#Id**, Nombre, Apellidos, Telfno, Direccion, Localidad, Cod_postal, Provincia, País, DNI)

TIPO_HABITACION (**#Id**, tipo, Precio_alta, Precio_baja)

HABITACIONES (**#Id**, Número, Id_tipo_habitacion, personas, estado)
TIPO_HABITACION

RESERVAS (**#Id**, Fecha_entrada, Fecha_salida, Personas, Precio, Late_checkout, Timestamp_entrada, Timestamp_salida, Id_cliente, Id_habitacion)
CLIENTES HABITACIONES

FACTURAS (**#Id**, Id_reserva, Número, Fecha, Forma_pago, Timestamp_pago)
RESERVAS

LINEASFACTURAS (**#Id**, Id_factura, Concepto, Cantidad, Precio, IVA)
FACTURAS

SERVICIOS (**#Id**, Nombre, Estado, Plazas, Precio)

RESERVA_SERVICIOS (**#Id**, Id_reserva, Id_servicio, Fecha, Hora_inicio, Hora_fin, Precio)
RESERVAS SERVICIOS

PERSONAL (**#Id**, Nombre, Apellidos, Telfno, Direccion, Localidad, Cod_postal, Provincia, País, DNI, Tipo)

INCIDENCIAS (**#Id**, Tipo, Descripcion, Id_personal, Fecha_notificacion, Id_habitacion, Fecha_resolucion)
PERSONAL HABITACION

LIMPIEZA_HABITACIONES (**#Id**, Id_personal, Id_reserva, Timestamp_limpieza)
PERSONAL RESERVA

USUARIOS (**#Id**, Usuario, Password, Id_personal)
PERSONAL

Fig. 4.3 Diagrama Relacional previsto

4.2. Análisis y diseño del sistema funcional

4.2.1. Diagrama de casos de uso

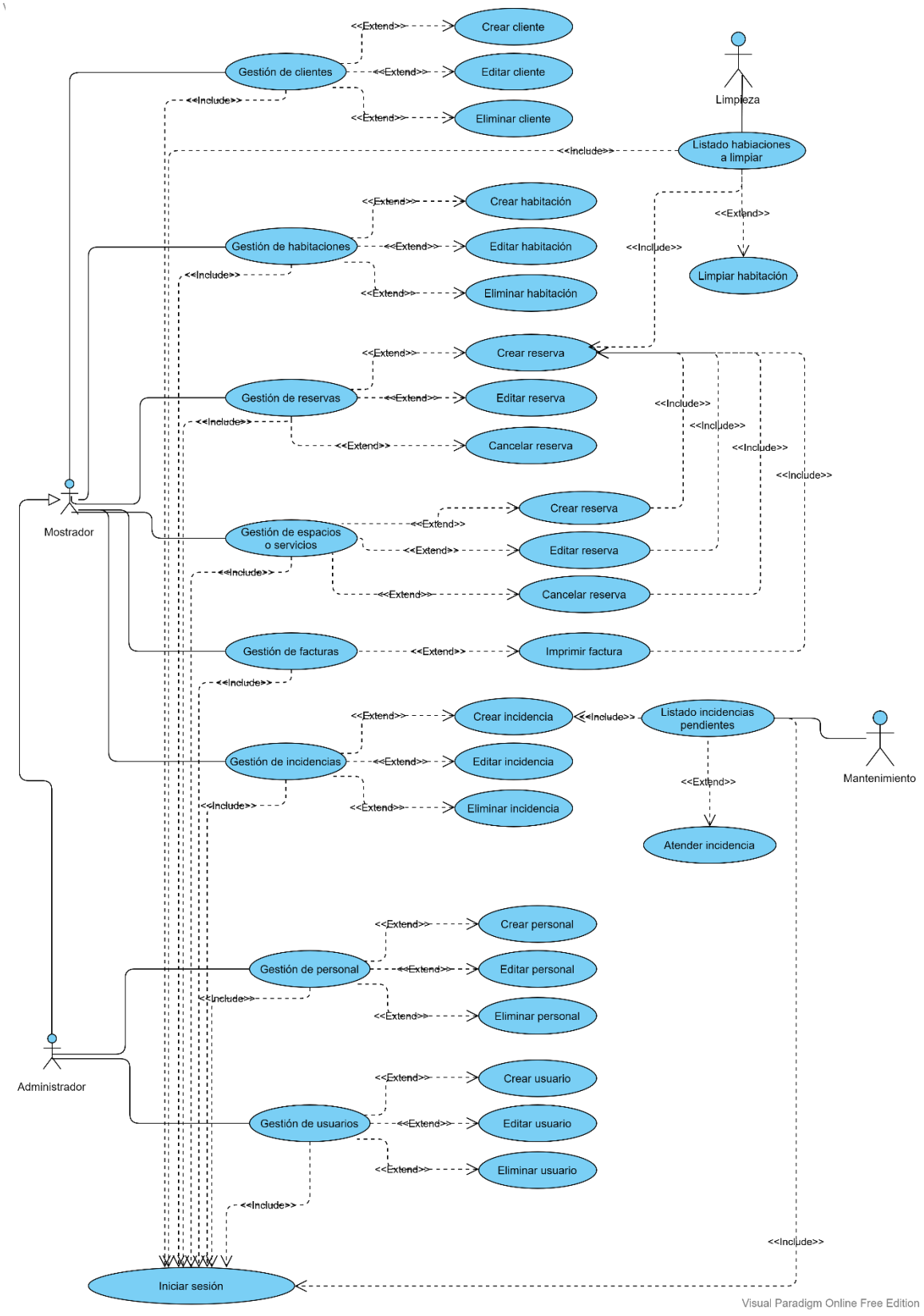


Fig. 4.4 Diagrama de casos de uso.

4.2.2. Diagrama de secuencia

Fijándose en el diagrama anterior, se puede ver que no hay mucho flujo en las interacciones, por tanto, el diagrama de secuencia se reduce a acciones que se puede decir que son mas o menos idénticas. Las categorizo en acciones que crean cosas, acciones que las editan, y acciones que eliminan. La mayor parte del proceso las lleva a cabo el personal del mostrador.

En las tres gestiones se establecen las tres mismas acciones. Por ello, solo se muestran las acciones de una gestión.

4.2.2.1. Creación de reservas (de habitaciones y de espacios), habitaciones, clientes, incidencias, personal y usuarios

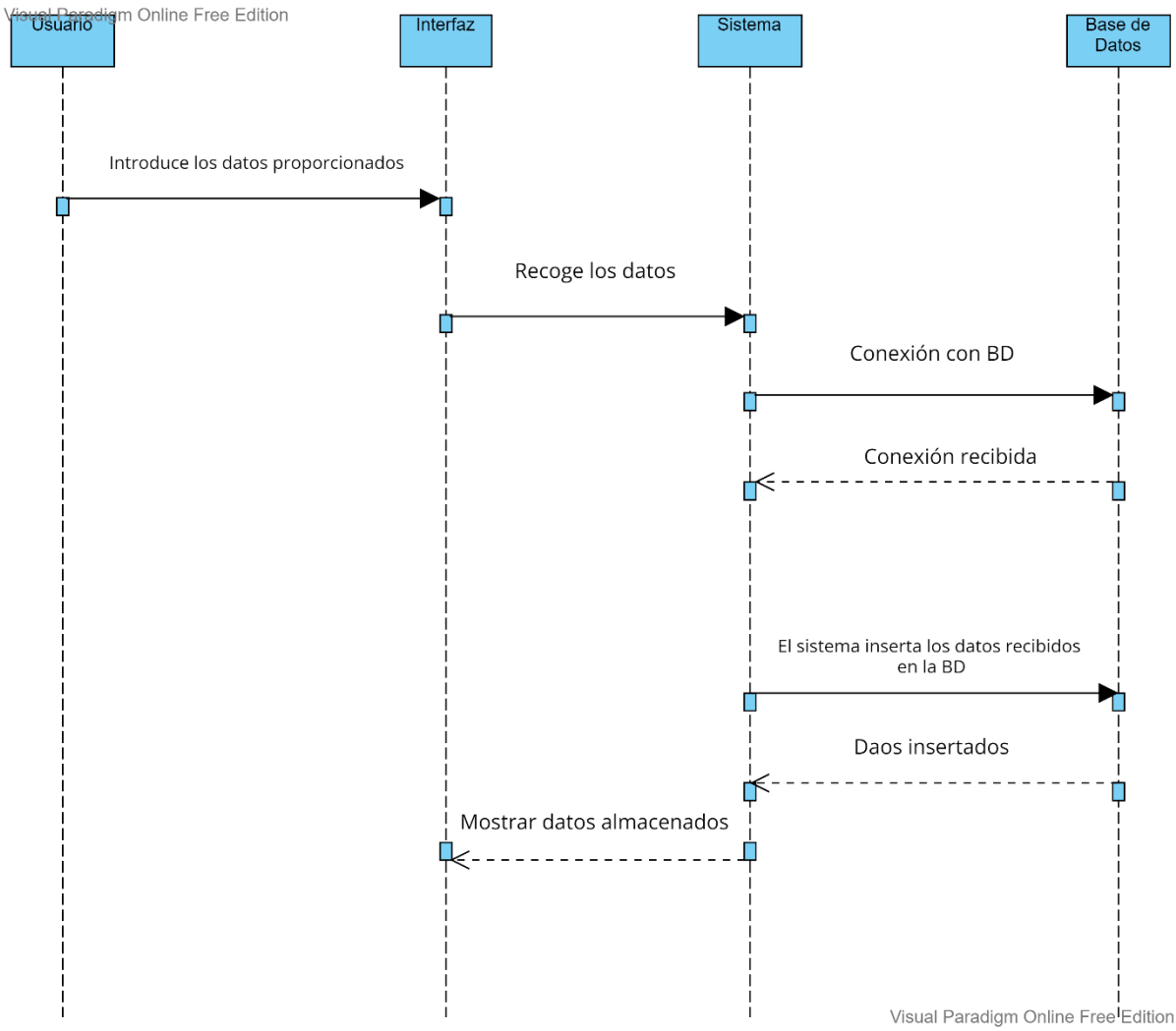


Fig. 4.5 Diagrama de secuencia: Creación

4.2.2.2. Edición de reservas (de habitaciones y de espacios), habitaciones, clientes, incidencias, personal y usuarios

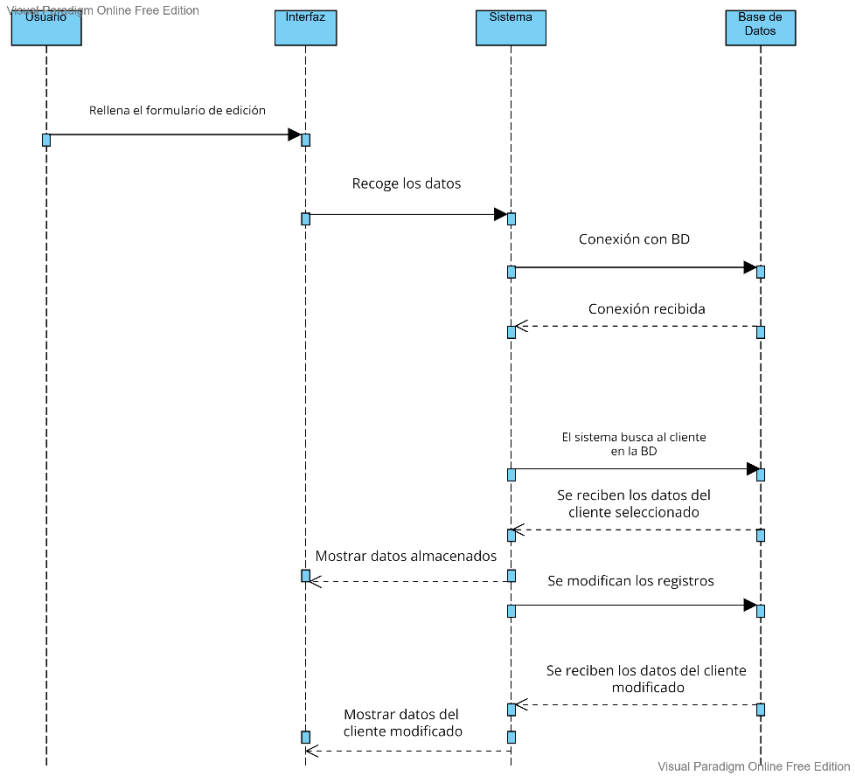


Fig. 4.6 Diagrama de secuencia: edición de cliente

4.2.2.3. Borrado de elementos

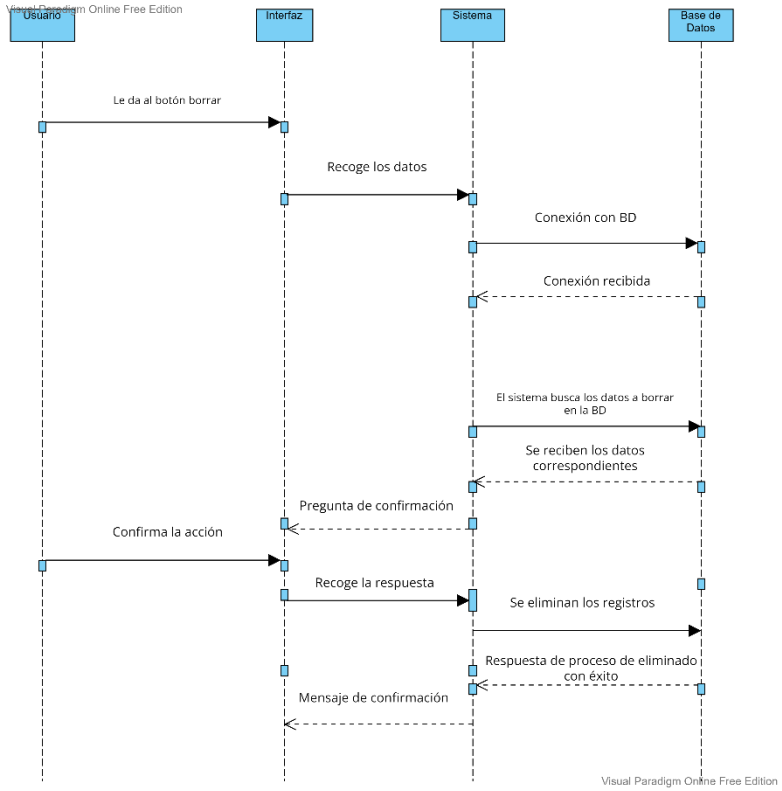


Fig. 4.7 Diagrama de secuencia: Borrado

4.2.2.4. Dashboard

La aplicación, como se ve más adelante, tiene un dashboard, que se muestra cuando alguien se loguea. Para que el tablero se muestre como se tiene que ver, el servidor debe llevar a cabo unas operaciones.

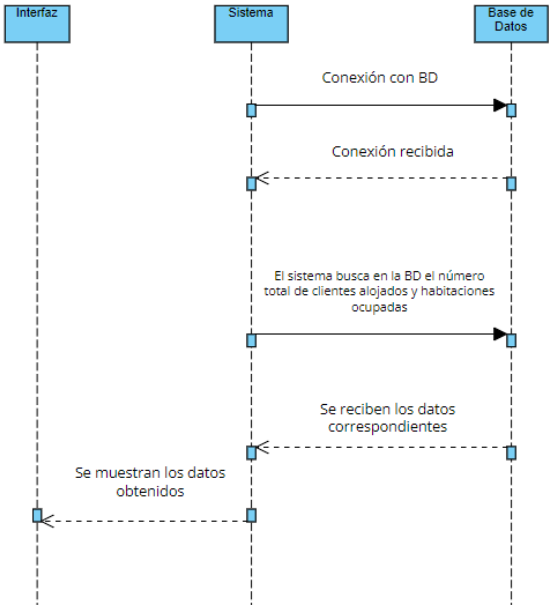


Fig 4.8 Datos mostrados en el Dashboard nada más loguearse.

Por supuesto, el registro también ofrece una interacción que debe atender el servidor.

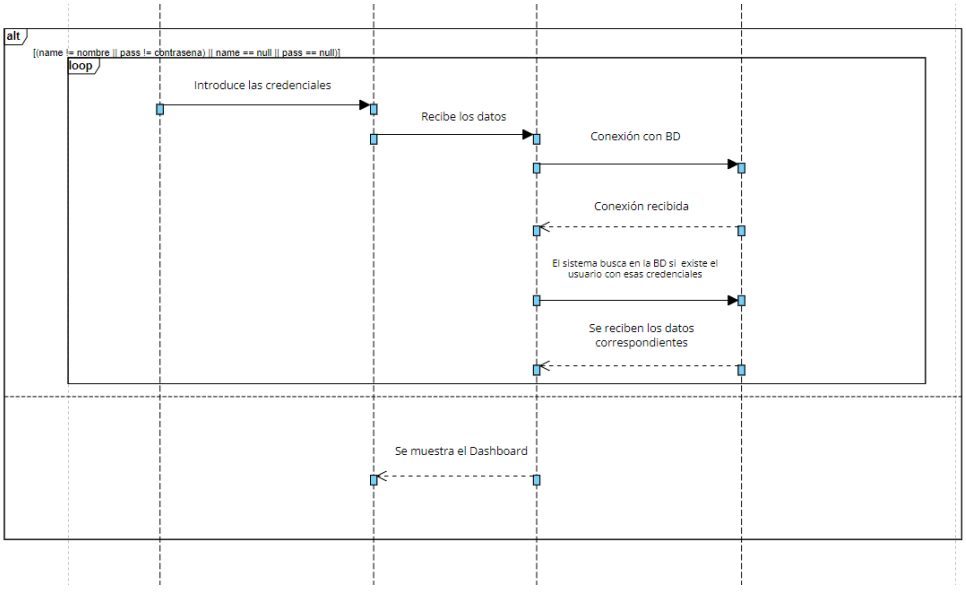


Fig 4.9 Proceso de registro

4.2.3. Diagrama de clases

Al hacer el proyecto en un framework, no conozco el funcionamiento de algunas clases que lleva predefinidas, y como de alguna de ellas heredan mis clases, como es el caso de Model, Controller o Migration, ponerlas no tiene mucho sentido, así que simplemente las menciono.

Así mismo, casi ninguna de las clases tiene atributos, como se ve en el diagrama, que sólo muestran los métodos y funciones que tienen cada una.

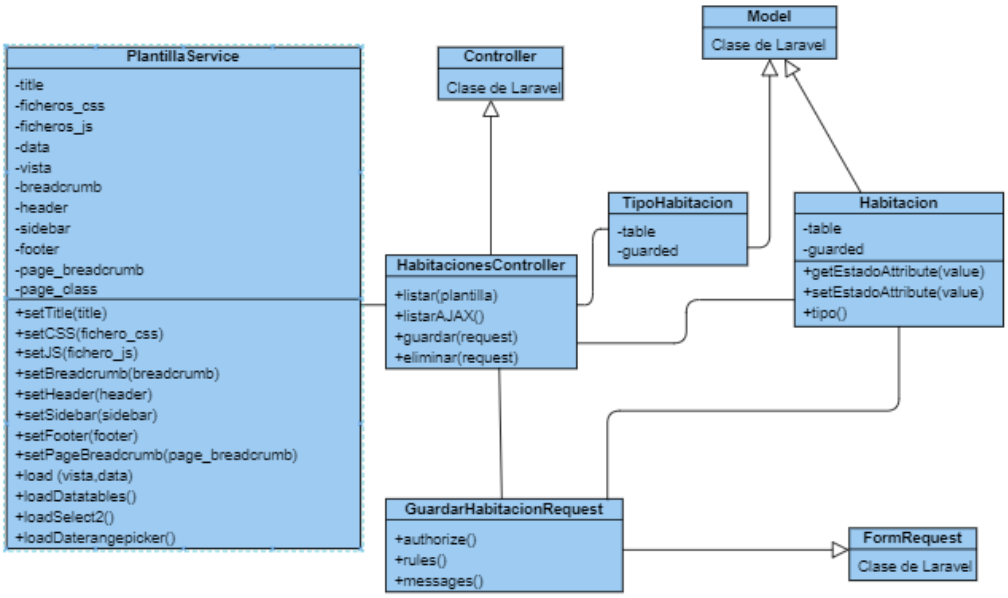


Fig 4.10 Diagrama de clases de HabitacionesController

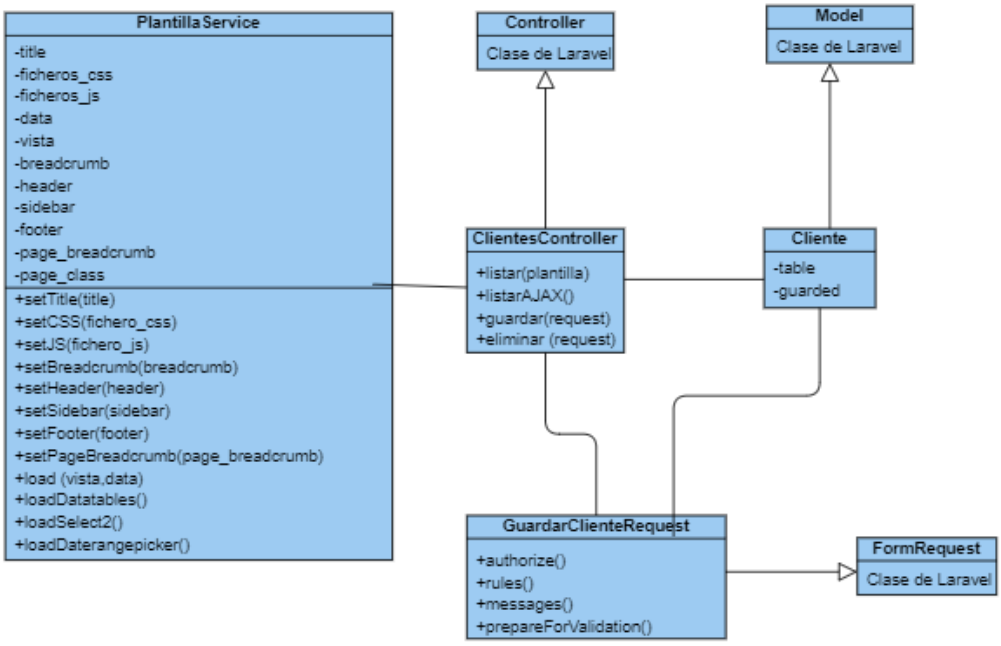


Fig 4.11 Diagrama de clases de ClientesController

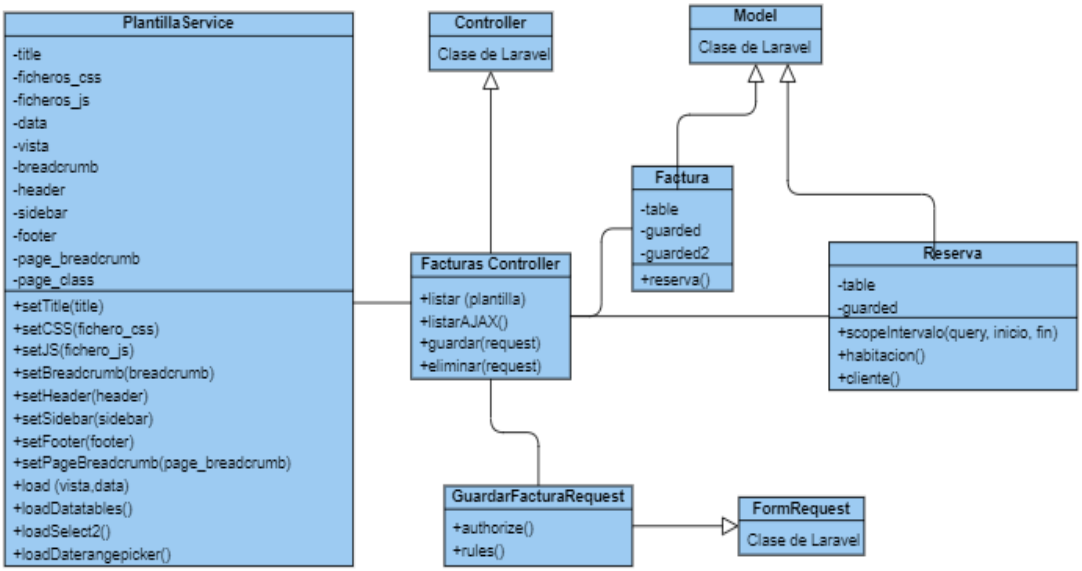


Fig 4.12 Diagrama de clases de FacturasController

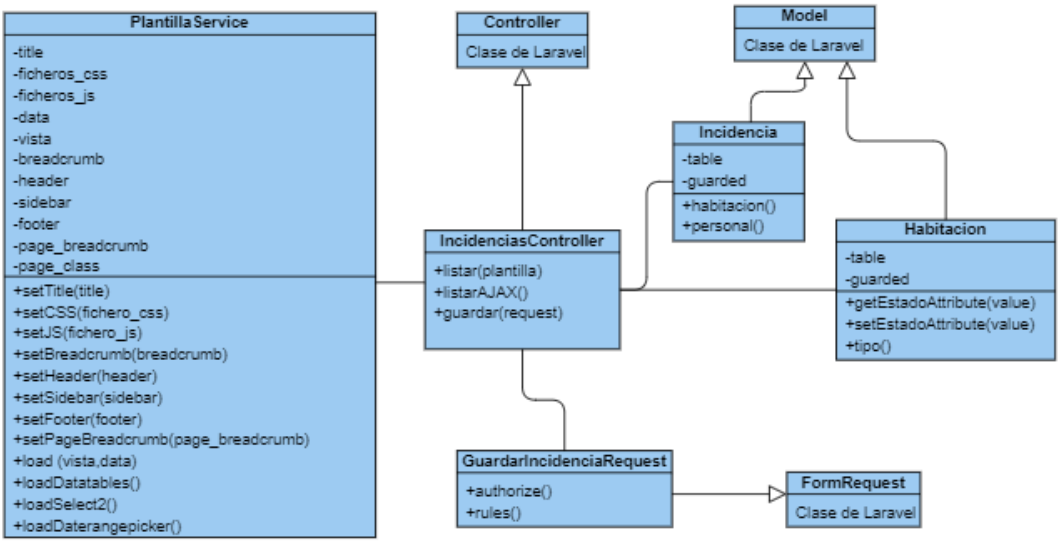


Fig 4.13 Diagrama de clases de IncidenciasController

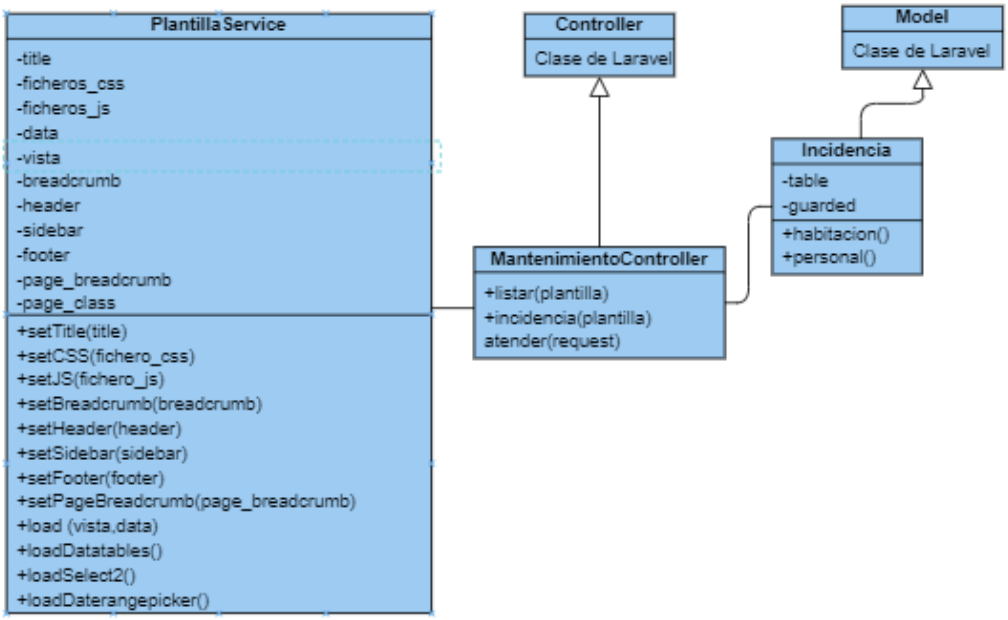


Fig 4.14 Diagrama de clases de MantenimientoController

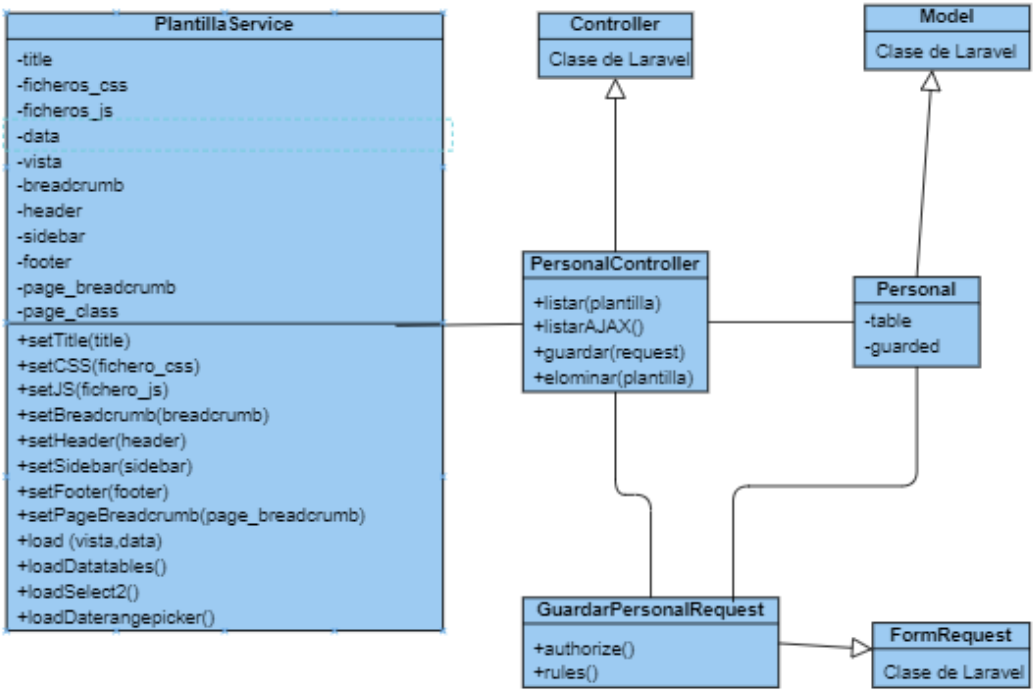


Fig 4.15 Diagrama de clases de PersonalController

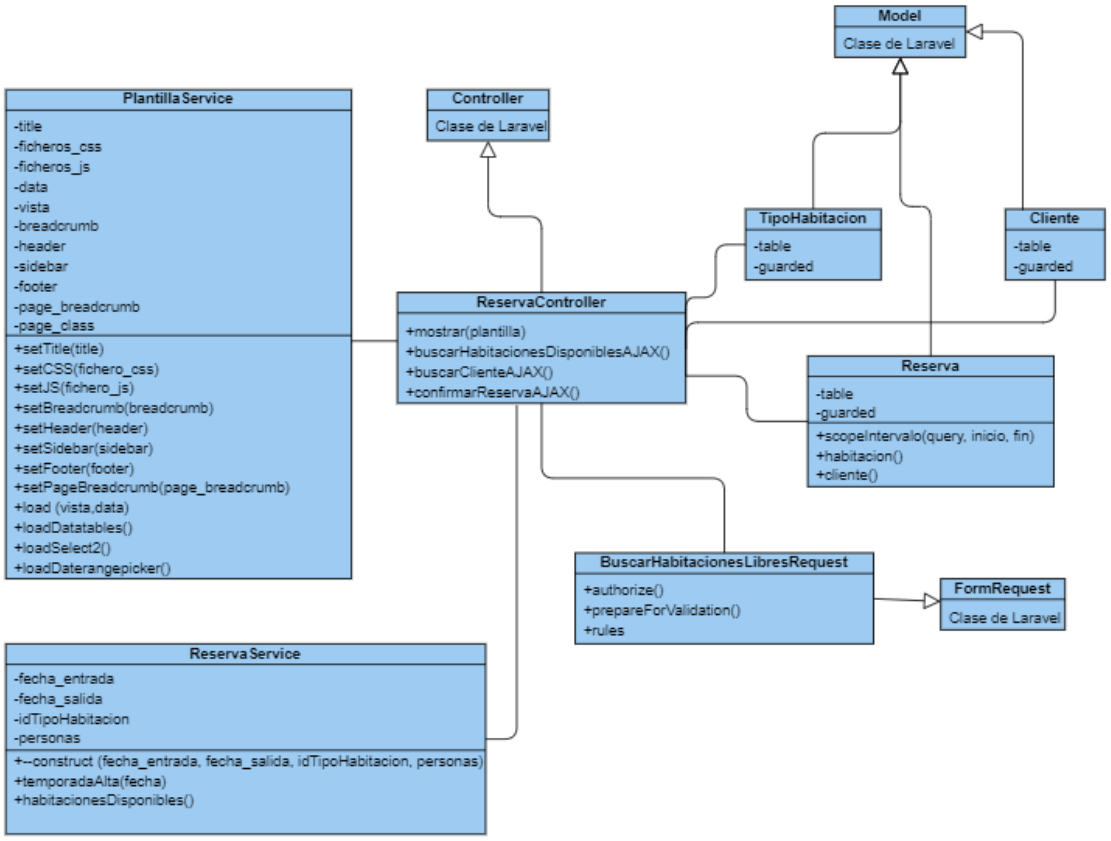


Fig 4.16 Diagrama de clases de ReservaController

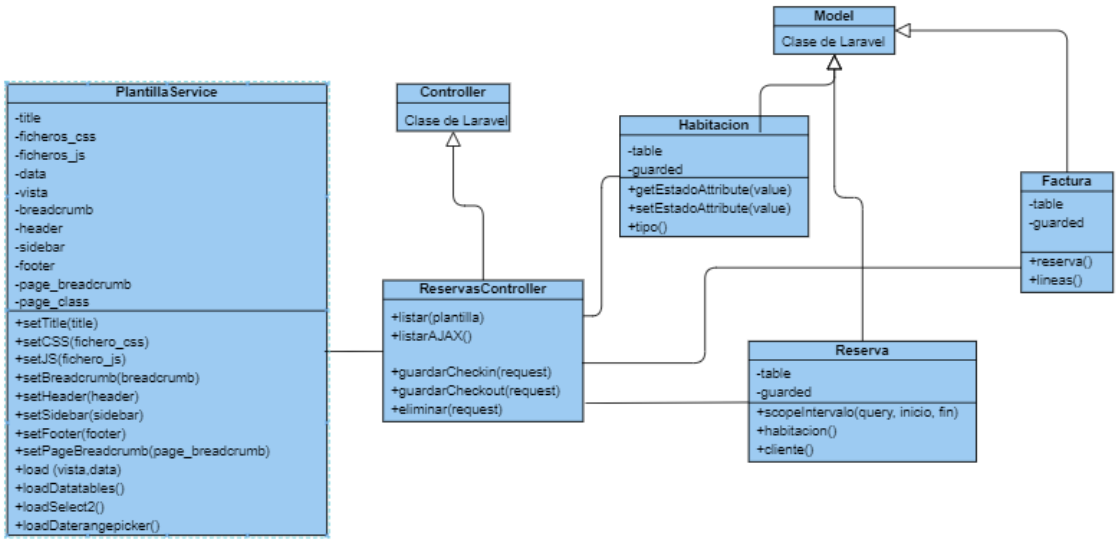


Fig 4.17 Diagrama de clases de ReservasController

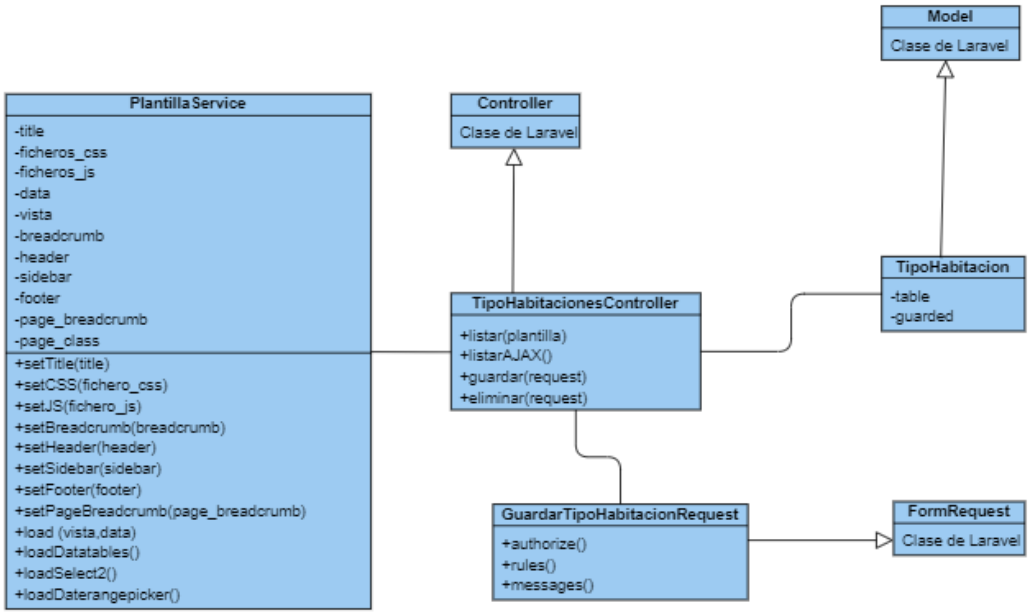


Fig 4.18 Diagrama de clases de TipoHabitacionesController

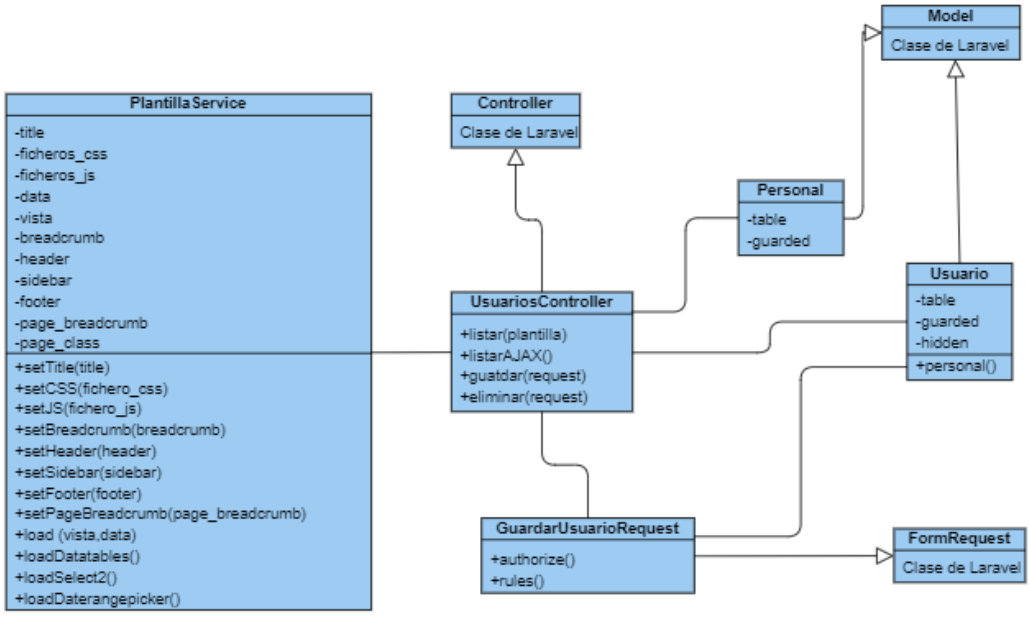


Fig 4.19 Diagrama de clases de UsuariosController

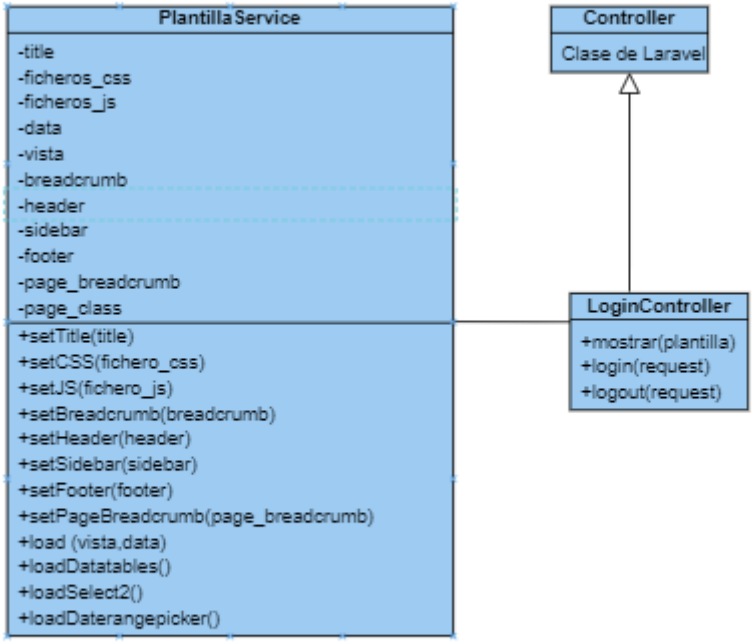


Fig 4.20 Diagrama de clases de LoginController

4.3. Análisis y diseño de la interfaz de usuario.

Ahora paso a mostrar los “mockups” o los bocetos de lo que es la interfaz del programa.

4.3.1. Aplicación web de escritorio

Como casi todos los mockups tienen un aspecto similar, se muestra los destacables y especiales:

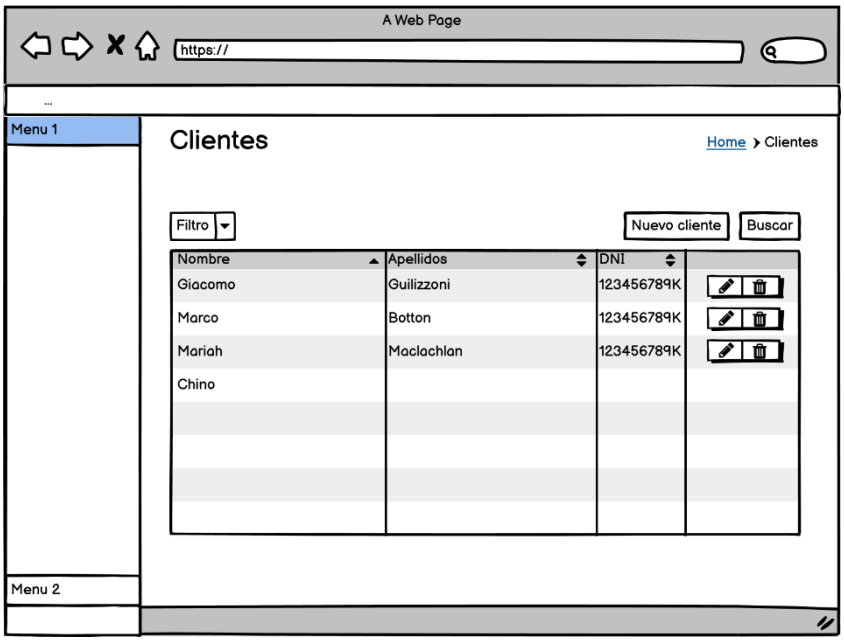


Fig. 4.21 Interfaz de listado de clientes.

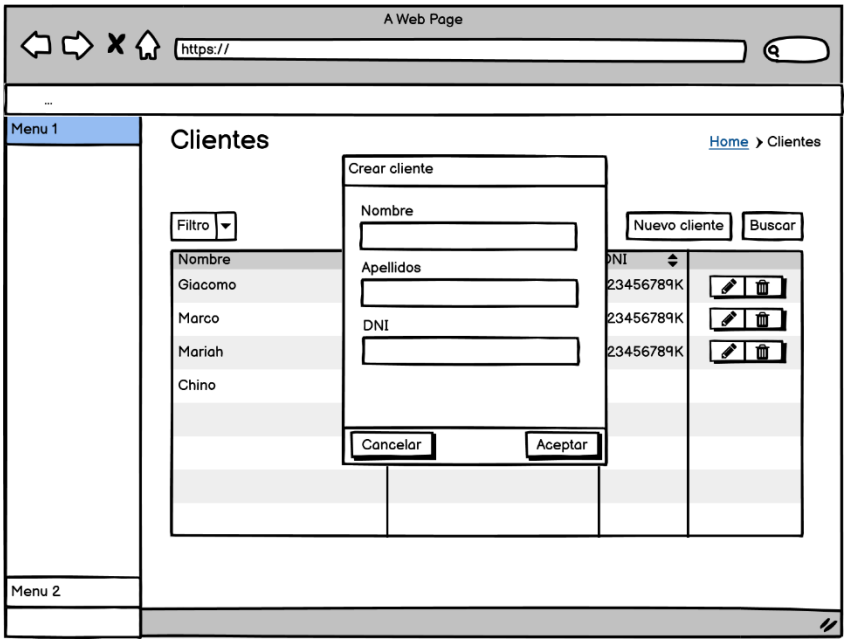


Fig. 4.22 Interfaz de edición / borrado de clientes.

Se abre un pop up donde se muestra el formulario correspondiente.

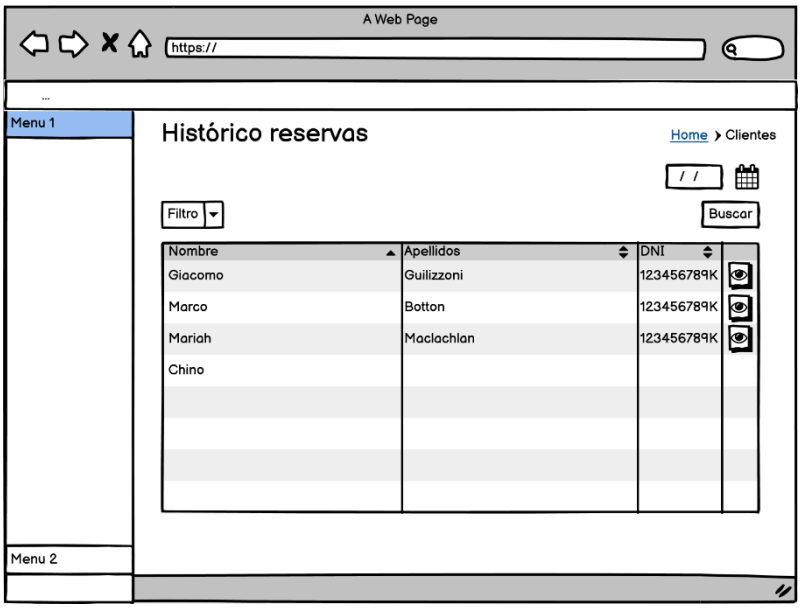


Fig. 4.23 Interfaz de listado de reservas

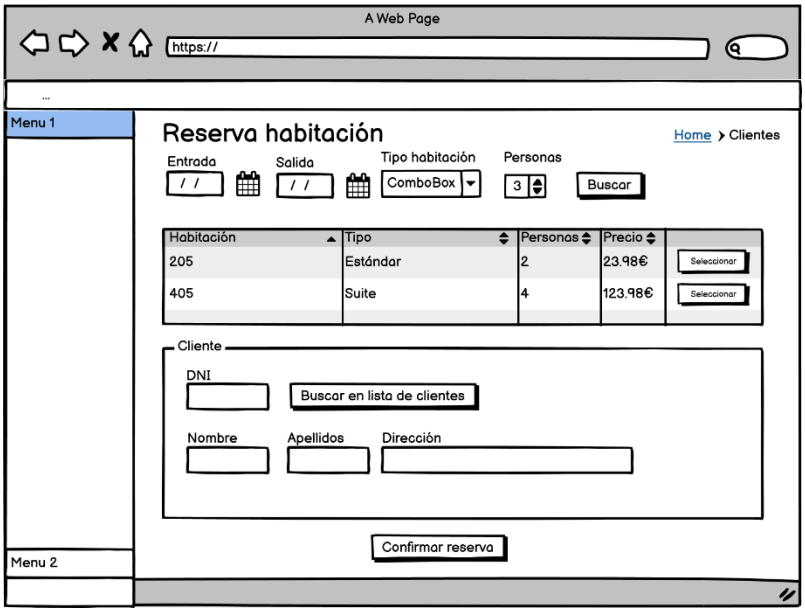


Fig. 4.24 Interfaz de reserva de una habitación.

En este caso la interfaz es más completa porque, además de añadir los datos del cliente sea la primera vez que se hospede o no, hay que obtener un listado de las habitaciones disponibles que cumplan los requerimientos solicitados por el cliente. Estos datos se utilizan después para generar la factura.

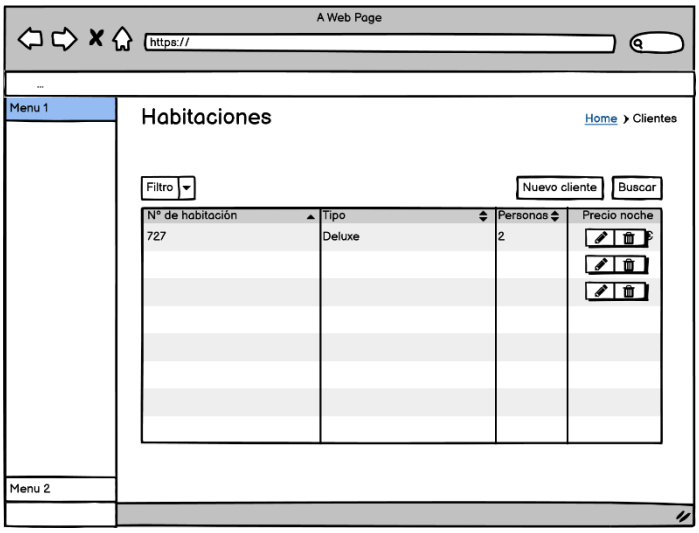


Fig. 4.25 Interfaz de listado de habitaciones.

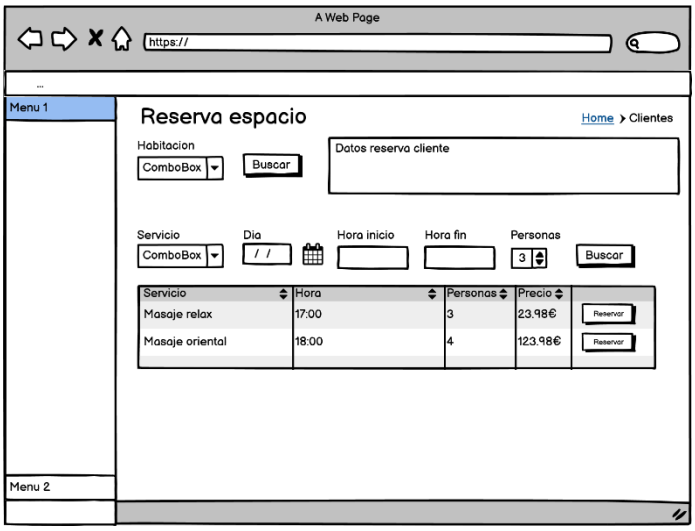


Fig. 4.26 Interfaz de reserva de los espacios y servicios del hotel.

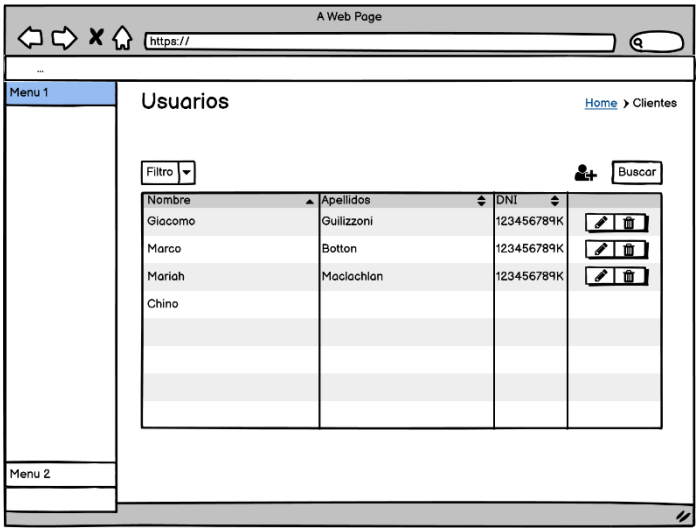


Fig. 4.27 Interfaz de lista de usuarios. Aquí el DNI sería el tipo de usuario.

A continuación, se muestra los únicos:

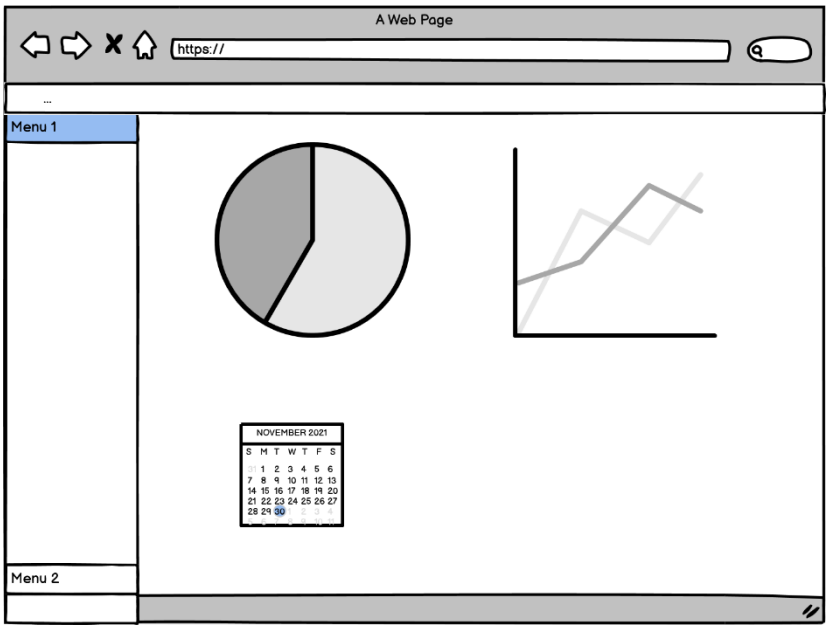


Fig. 4.28 Dashboard de la aplicación. Es lo que se ve tras autentificarse con éxito los que tienen acceso a la aplicación de escritorio.

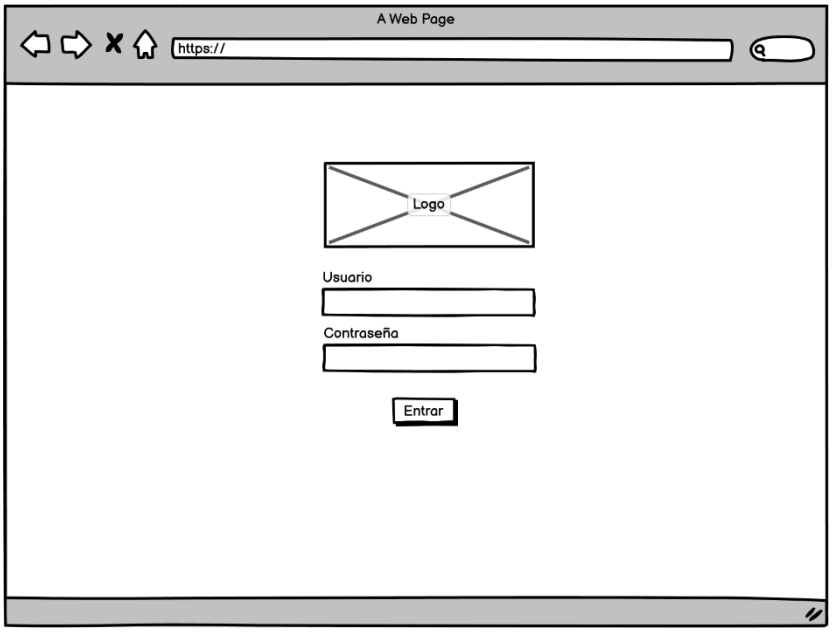


Fig. 4.29 Interfaz del login de la aplicación. Esta página es la que se ve al acceder a la URL.

4.3.2. Aplicación móvil

Esta versión de la aplicación la usan los empleados encargados del mantenimiento y limpieza del hotel.

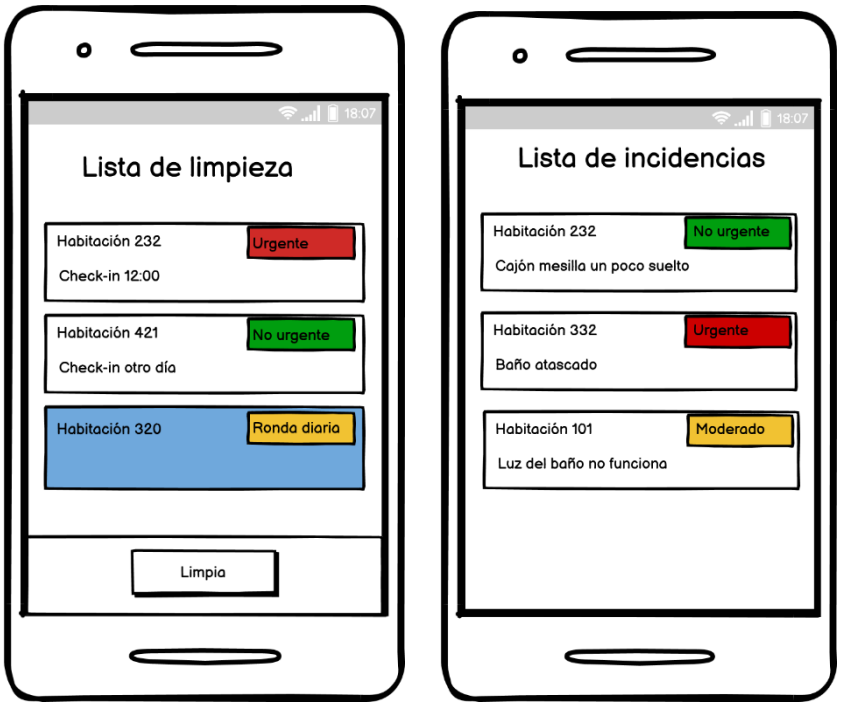


Fig. 4.30 y 4.31 Dashboards del personal de limpieza y mantenimiento, respectivamente.

Fig. 4.32 y 4.33 Ficha de incidencia del personal de limpieza y mantenimiento, respectivamente.

4.4. Diseño de la arquitectura de la aplicación

4.4.1. Tecnologías/Herramientas usadas y descripción de las mismas

Para el proyecto se ha utilizado el lenguaje de programación PHP por ser uno de los más usados y por su facilidad para conectarse a la base de datos. Además, el rendimiento es muy

superior, sobre todo a partir de la versión 7. Es compatible con la mayoría de las bases de datos y tiene una enorme comunidad.

Para mayor comodidad se ha usado el framework de Laravel, basado también en Symphony, y que también es muy utilizado. Laravel permite escribir menos código, ya que posee muchas funcionalidades que automatizan el proceso. Es simple, pero potente, y tiene una interfaz elegante y agradable de usar.

Su mayor baza es el sistema de ruteo (RESTfull) que se sirve de dependencias y controladores. De esta forma, se puede juntar todo el código en un solo archivo y en cada ruta "mandar" a que se ejecute el código correspondiente, sin tener que buscar en las carpetas el archivo que contiene el código necesitado.

Además, se implementan otras tecnologías, librerías y plugins:

- **HTML** (Hypertext Markup Laguage o Lenguaje de Marcas de Hipertexto): Es el componente más básico de la Web. Utiliza "marcas" para etiquetar texto, imágenes y otros contenidos para visualizarlo en el navegador, ya que éste es capaz de interpretarlo. Hipertexto hace referencia a los links o enlaces que conectan páginas web entre sí. La última versión y la más utilizada es HTML5.
- **CSS** (Cascade Style Sheets o Hojas de Estilo en Cascada): Describe el estilo que tienen las páginas web. Es en cascada porque el estilo de una etiqueta, por ejemplo, puede influir en otra, así como heredarla. La versión oficial y más reciente es la CSS3.
- **JavaScript**: Es un lenguaje de programación del lado del cliente, es decir, interpretado. Se suele conocer por se un lenguaje basado en scripts, aunque también se basa en prototipos, dinámico, ligero y con soporte para programación orientada a objetos (POO).
- **AJAX** (JavaScript asíncrono + XML): Más que una tecnología propia, es un modo de usar otras. Su objetivo es mejorar el rendimiento, ya que, al ser asíncrono, se puede crear páginas web que pueden ser actualizadas sin necesidad de volver a cargar la página, haciendo la web o la aplicación web más rápida y con mejor respuesta a las acciones del cliente.
- **Jquery**: Es una librería de JavaScript muy utilizada que trata de simplificar la manipulación del Modelo de Objetos del Documento o en inglés Document Object Model (DOM), llamadas AJAX y manejos de eventos.
- **Moment**: Otra librería útil para analizar, validar, manipular y mostrar fechas y horas en JavaScript.
- **Bootstrap**: Es un framework CSS desarrollado por Twitter en 2010. Inicialmente, se llamaba Twitter Blueprint, pero en 2011 pasó a ser de código abierto y su nombre cambió a Bootstrap. Es el framework más usado a día de hoy. Proporciona interactividad en las páginas, de tal forma que ofrece muchos componentes como menús de navegación, controles de página, etc. Pero su verdadero potencial es que es muy personalizable y sobre todo permite la construcción de webs responsive, es decir, adaptadas a todo tipo de dispositivos, incluidos móviles. Su versión más reciente, la 5.1, es la utilizada para este proyecto.
- **Datatables**: Es un plugin de Jquery. Según la web oficial es una herramienta

altamente flexible, que agrega varias funciones a HTML, tales como paginación, búsqueda instantánea, ordenación de columnas y más. De código abierto, este plugin tiene la posibilidad de ampliarse con extensiones. Es responsive (adaptable a móviles).

- **Select2**: Es otro plugin que permite insertar un cuadro de selección (lista) personalizable y se le pueden complementar con extensiones. Usa AJAX para buscar en la lista desplegable, en caso de ser necesario. Select2 tiene su propio repositorio en GitHub.
- **Daterangepicker**: Este plugin inserta un espacio para uno o varios calendarios emergentes, permitiendo establecer fechas, horas, o un rango de fechas. Se creó para informes en Improvely, un servicio online de marketing.
- **Laravel Validator**: Un validador de formularios para Laravel disponible en Github. Valida, entre otras cosas si el número de DNI puede ser válido o no (si la letra corresponde con los números).
- **Font Awesone**: En esta web se pueden encontrar iconos tanto gratis como de pago con sus correspondientes códigos HTML para incrustarlos directamente en el proyecto.
- **PHPMyAdmin**: Programa que proporciona una interfaz donde es posible ver, editar y manipular bases de datos.

4.4.2. Arquitectura de componentes de la aplicación

La estructura que mi aplicación web sigue el modelo de capas, las cuales expongo a continuación:

Capa cliente. Se usa como cliente el navegador, teniendo como propósito que se vea en la mayoría de los navegadores: Google Chrome, Microsoft Edge, Opera, Firefox, etc.

Capa media. Comprende la presentación, que utilizo un servidor web en local (Apache). La lógica de negocio la lleva el servidor de aplicaciones, en mi caso es Laravel que usa PHP. Mediante AJAX se comunica con JavaScript y éste se comunica con la capa cliente.

Capa de datos. Como gestor de base de datos uso el MariaDB 10.4.14, que viene por defecto con XAMPP.

Transcurso

El usuario accede a la aplicación donde se le muestran las distintas interfaces (HTML en la view y JS). Cuando realiza una acción como hacer clic en un botón, el evento lo recoge JavaScript. Si es algo que no requiere el comportamiento del servidor, se hacen las funciones oportunas en el propio JavaScript y el usuario visualiza los cambios.

Sin embargo, hay determinadas acciones que requieren al servidor para operar. Como JavaScript no puede acceder directamente a los datos de la base de datos, este se comunica con la clase “Controller” a través de una llamada AJAX. El controlador procesa y valida los datos de la petición a través de una clase “Request” que responderá en caso de error. Si no hay error, se realiza el comportamiento necesario en el controlador, usando las clases “Model” que son las encargadas de comunicarse directamente con la BBDD. Si las acciones a realizar son complejas

se usan clases adicionales propias que hemos llamado “Services” para que lleven a cabo dichas acciones. Por último, se devuelven los datos que correspondan al cliente para que este muestre los cambios al usuario.

DOCUMENTO DE IMPLEMENTACIÓN, PRUEBAS E IMPLANTACIÓN DEL SISTEMA

5. Documento de pruebas funcionales

5.1. Implementación

5.1.1. Scripts SQL

Uso Laravel como framework de PHP a la hora de programar, y, debido a su funcionamiento, construyo las tablas de la base de datos con unos archivos que se llaman migraciones. Como no trabajo con SQL, en su lugar pongo las migraciones como creates y drops, y los select, insert y update en la sintaxis que Laravel usa para permitir manipular la base de datos, llamada **Eloquent.** Lo que pongo a continuación son capturas con el código de los ficheros escritos en PHPStorm.

5.1.1.1. Migraciones (CREATE y DROP)

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateClientesTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('clientes', function (Blueprint $table) {
            $table->id();
            $table->string('nombre', 100)->nullable();
            $table->string('apellidos', 100)->nullable();
            $table->string('telefono', 20)->nullable();
            $table->string('direccion', 255)->nullable();
            $table->string('localidad', 100)->nullable();
            $table->string('cod_postal', 100)->nullable();
            $table->string('provincia', 100)->nullable();
            $table->string('pais', 100)->nullable();
            $table->string('dni', 20)->unique()->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('clientes');
    }
}
```

Fig. 5.1 Migración de la tabla Clientes.

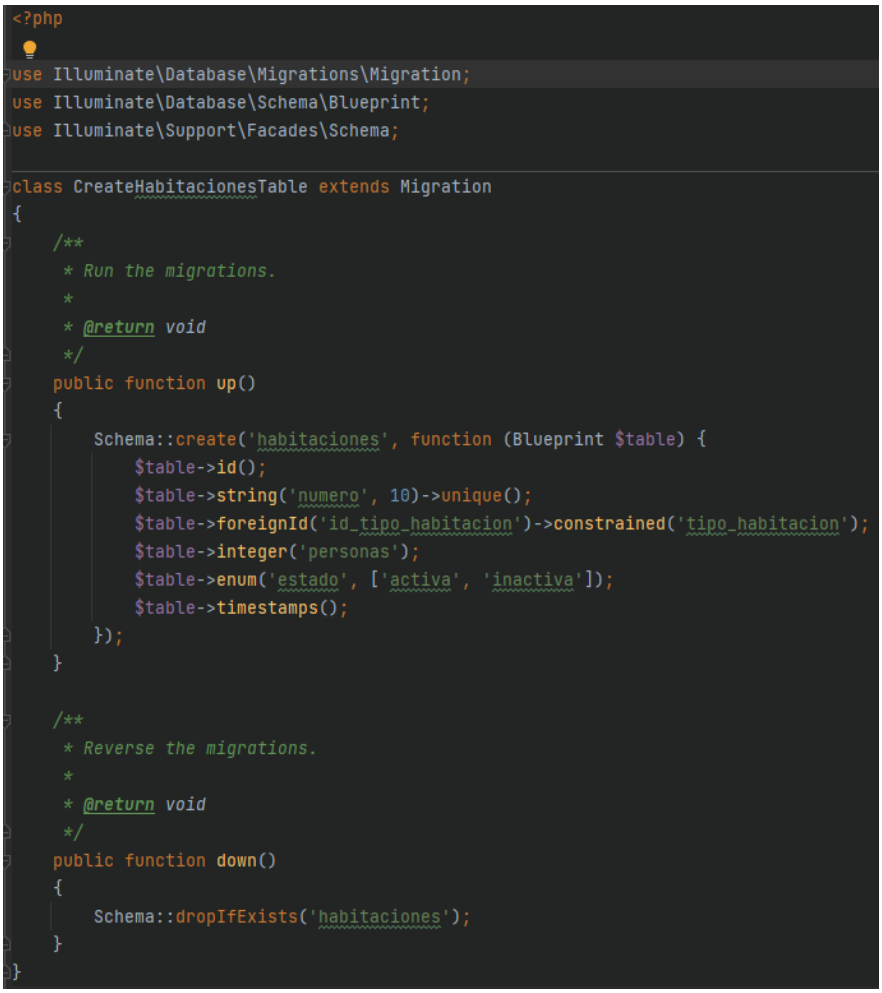
```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateTipoHabitacionTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('tipo_habitacion', function (Blueprint $table) {
            $table->id();
            $table->string('tipo', 50)->nullable();
            $table->double('precio_alta')->nullable();
            $table->double('precio_baja')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('tipo_habitacion');
    }
}
```

Fig. 5.2 Migración de la tabla TipoHabitacion



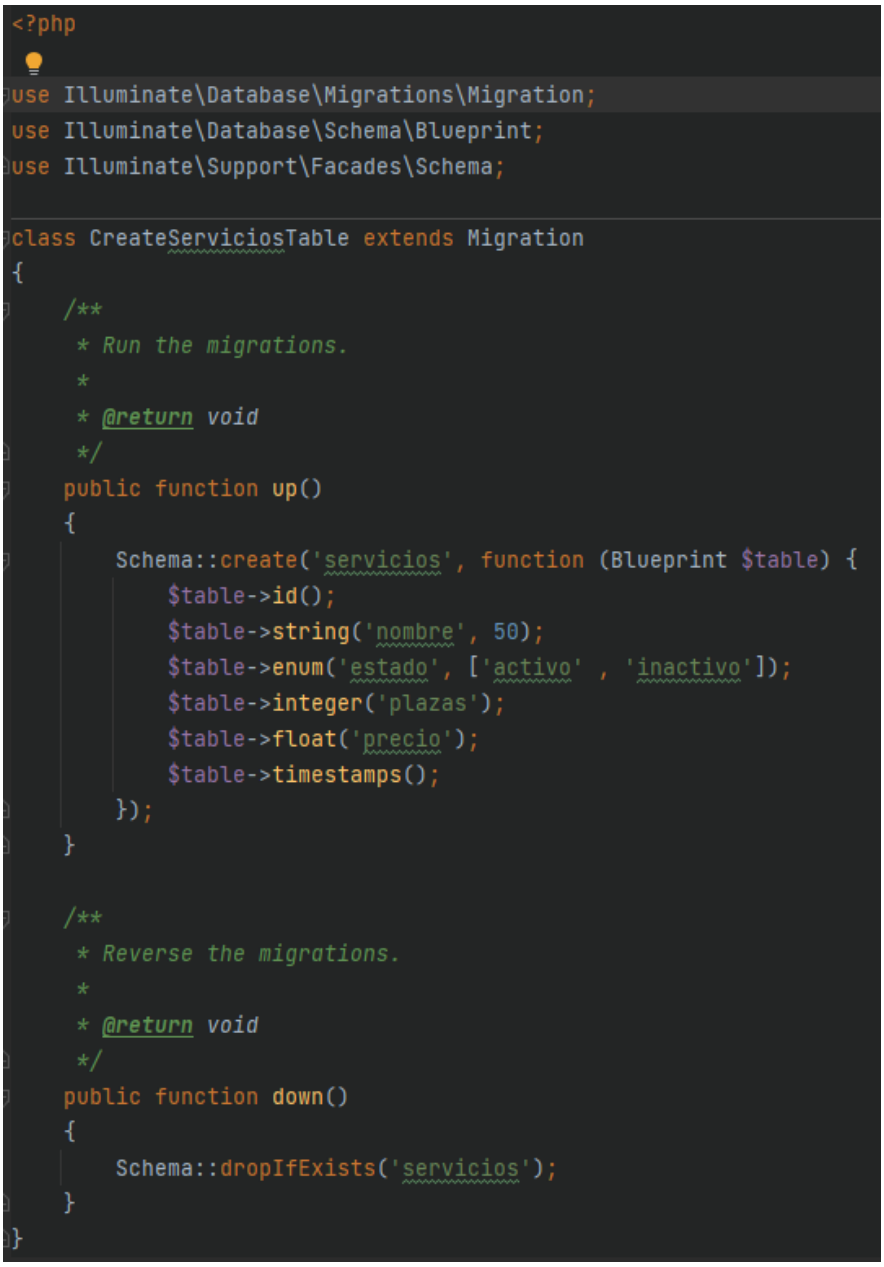
```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateHabitacionesTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('habitaciones', function (Blueprint $table) {
            $table->id();
            $table->string('numero', 10)->unique();
            $table->foreignId('id_tipo_habitacion')->constrained('tipo_habitacion');
            $table->integer('personas');
            $table->enum('estado', ['activa', 'inactiva']);
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('habitaciones');
    }
}
```

Fig. 5.3 Migración de la tabla Habitaciones



```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateServiciosTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('servicios', function (Blueprint $table) {
            $table->id();
            $table->string('nombre', 50);
            $table->enum('estado', ['activo' , 'inactivo']);
            $table->integer('plazas');
            $table->float('precio');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('servicios');
    }
}
```

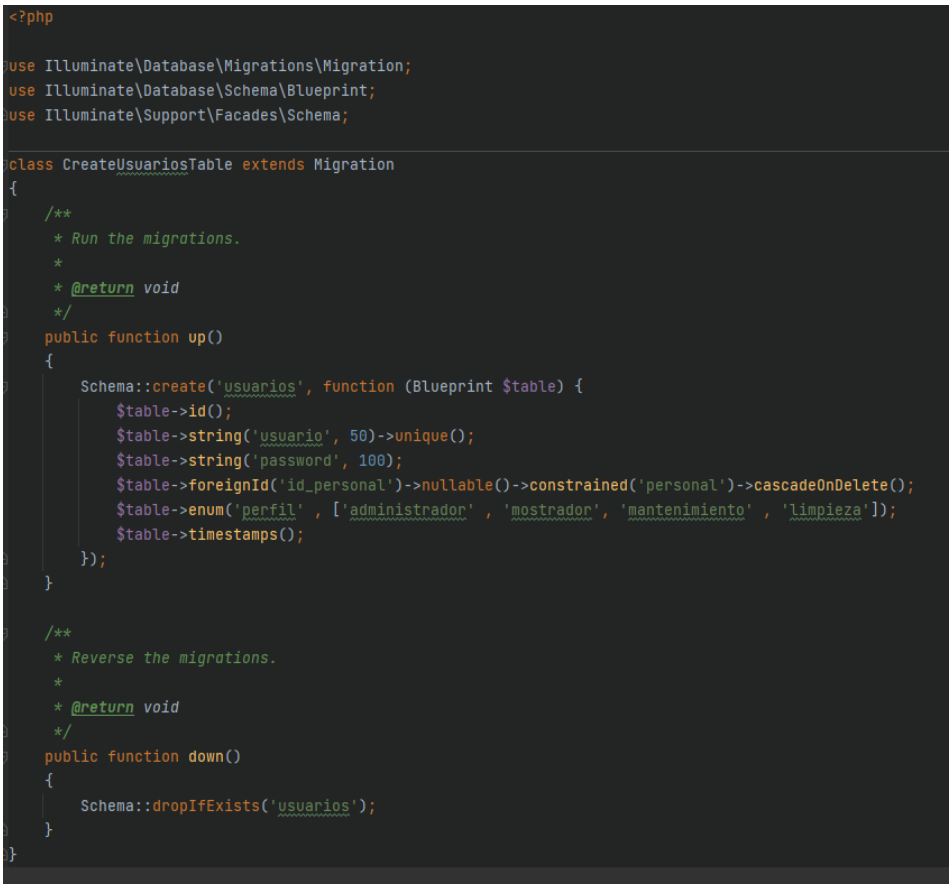
5.4 Migración de la tabla Servicios

```php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreatePersonalTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('personal', function (Blueprint $table) {
            $table->id();
            $table->string('nombre', 100)->nullable();
            $table->string('apellidos', 100)->nullable();
            $table->string('telefono', 20)->nullable();
            $table->string('direccion', 255)->nullable();
            $table->string('localidad', 100)->nullable();
            $table->string('cod_postal', 100)->nullable();
            $table->string('provincia', 100)->nullable();
            $table->string('pais', 100)->nullable();
            $table->string('dni', 20)->unique()->nullable();
            $table->string('tipo', 20);
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('personal');
    }
}
```

Fig 5.5 Migración de la tabla Personal



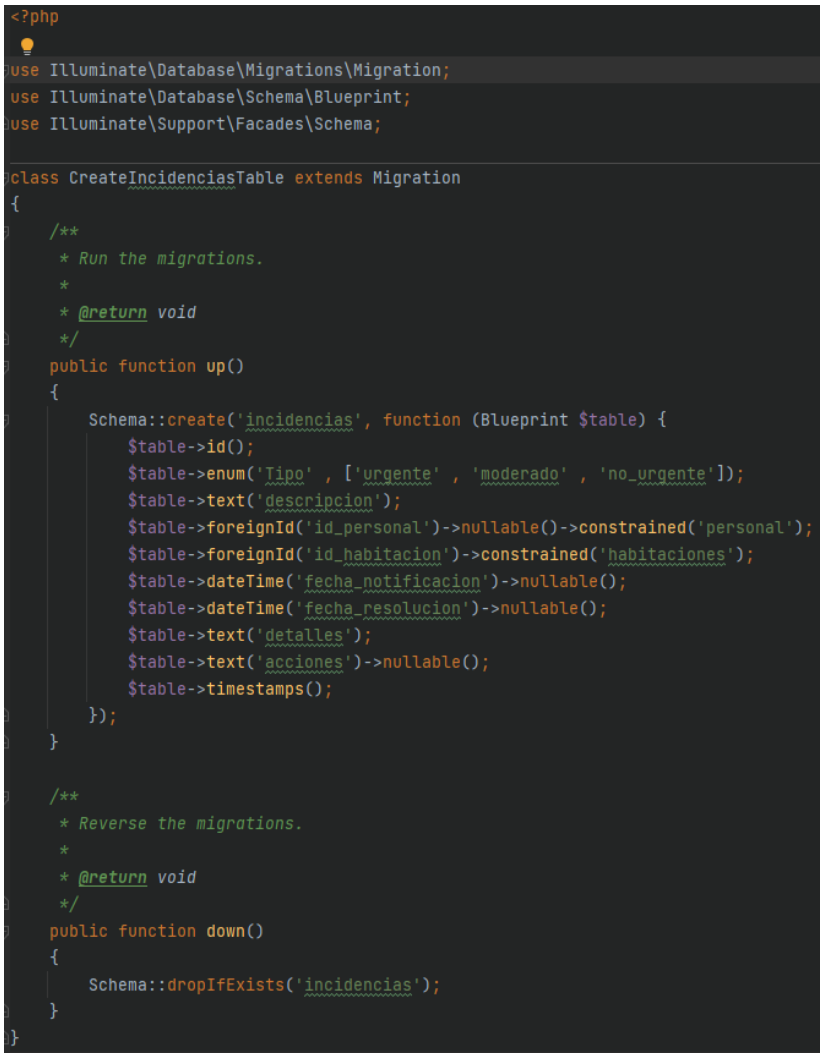
```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateUsuariosTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('usuarios', function (Blueprint $table) {
            $table->id();
            $table->string('usuario', 50)->unique();
            $table->string('password', 100);
            $table->foreignId('id_personal')->nullable()->constrained('personal')->cascadeOnDelete();
            $table->enum('perfil' , ['administrador' , 'mostrador', 'mantenimiento' , 'limpieza']);
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('usuarios');
    }
}
```

Fig 5.6 Migración de la tabla Usuarios.



```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateIncidenciasTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('incidencias', function (Blueprint $table) {
            $table->id();
            $table->enum('Tipo' , ['urgente' , 'moderado' , 'no_urgente']);
            $table->text('descripcion');
            $table->foreignId('id_personal')->nullable()->constrained('personal');
            $table->foreignId('id_habitacion')->constrained('habitaciones');
            $table->dateTime('fecha_notificacion')->nullable();
            $table->dateTime('fecha_resolucion')->nullable();
            $table->text('detalles');
            $table->text('acciones')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('incidencias');
    }
}
```

Fig 5.7 Migración de la tabla Incidencias

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateReservasTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('reservas', function (Blueprint $table) {
            $table->id();
            $table->date('fecha_entrada');
            $table->date('fecha_salida');
            $table->integer('personas');
            $table->integer('precio');
            $table->boolean('late_checkout');
            $table->timestamp('timestamp_salida')->nullable();
            $table->timestamp('timestamp_entrada')->nullable();
            $table->foreignId('id_cliente')->constrained('clientes');
            $table->foreignId('id_habitacion')->constrained('habitaciones');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('reservas');
    }
}
```

Fig 5.8 Migración de la tabla Reservas

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateFacturasTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('facturas', function (Blueprint $table) {
            $table->id();
            $table->foreignId('id_reserva')->constrained('reservas');
            $table->string('numero');
            $table->date('fecha');
            $table->enum('forma_pago' , ['Efectivo' , 'Tarjeta' , 'Transferencia' , 'Cheque'])->nullable();
            $table->timestamp('timestamp_pago')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('facturas');
    }
}
```

Fig 5.9 Migración de la tabla Facturas

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateLineasfacturasTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('lineasfacturas', function (Blueprint $table) {
            $table->id();
            $table->foreignId('id_factura')->constrained('facturas')->cascadeOnDelete();
            $table->string('concepto', 500);
            $table->integer('cantidad');
            $table->double('precio');
            $table->double('base_imponible');
            $table->double('iva');
            $table->double('subtotal');
            $table->timestamps();
        }); // no migrada
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('lineasfacturas');
    }
}
```

Fig 5.10 Migración de la tabla Lineasfacturas

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateReservaServiciosTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('reserva_servicios', function (Blueprint $table) {
            $table->id();
            $table->foreignId('id_reserva')->constrained('reservas');
            $table->foreignId('id_servicio')->constrained('servicios');
            $table->date('fecha');
            $table->dateTime('hora_inicio');
            $table->dateTime('hora_fin');
            $table->integer('precio');
            $table->timestamps();
        }); // no migrada
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('reserva_servicios');
    }
}
```

Fig 5.11 Migración de la tabla ReservaServicios

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateLimpiezaHabitacionesTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('limpieza_habitaciones', function (Blueprint $table) {
            $table->id();
            $table->foreignId('id_personal')->constrained('personal');
            $table->foreignId('id_reserva')->constrained('reservas');
            $table->timestamp('timestamp_limpieza');
            $table->timestamps();
        }); //no migrada
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('limpieza_habitaciones');
    }
}
```

Fig 5.12 Migración de la tabla LimpiezaHabitaciones

5.1.1.2. Inserciones, actualizaciones y selecciones (INSERT, UPDATE y SELECT)

Para mis pruebas, creo mucho contenido y después lo modifico. Además, tengo la necesidad algunas veces de coger algunos datos y para ello, consulto la base de datos. No hay que confundir el create de inserción con el create de creación de la tabla. Aquí pongo unos ejemplos:

```php
$array = $request->validated();
$array ['numero'] = $nueva_factura;
Factura::create($array());
```

Fig 5.1 Un modo de insert.

Aquí inserto en la tabla Facturas (INSERT) una nueva factura cuyos datos, previamente metidos en la variable request, son validados y metidos en un array, también añado al array el número de factura, previamente asignado, y finalmente inserto los datos del array en la tabla.

```php
//crear
Habitacion::create($request->validated());
```

Fig 5.2 Otro modo de insert

Aquí inserto los datos validados de la variable request (INSERT). En este caso, no tengo que modificar nada, por tanto no necesito crear ningun array.

```php
Habitacion::find($request->id)->update($request->validated());
```

Fig 5.3 Update

Aquí actualizo la tabla Habitaciones (UPDATE) buscando el id pasado mediante AJAX, con los

datos metidos en la variable request.

```
$ultima_factura = (Factura::select('numero')->where('created_at', Factura::max('created_at'))->first());
```

Fig 5.4 Select

Aquí hago SELECT de la tabla Facturas; el valor que cojo es la última fecha que hay en la tabla, para hallar la factura más reciente.

```
Habitacion::destroy($request->id);
```

Fig 5.5 Delete

Aquí borro aquella fila de la tabla Habitaciones (DELETE) cuyo id es el que se ha enviado a la variable request.

Los datos, como se ve, siempre vienen en la variable request.

5.1.2. Gestión de cambios

Para registrar y llevar un control de cada cambio en el código, se usa Github como sistema de control de versiones. Si bien es cierto que en algunos commit no se sube al repositorio web, en los últimos cambios se van a subir a Github con lo que el cliente tiene a su disposición el código en la plataforma.

Además, se baraja la posibilidad de llevar un registro en la plataforma Trello, aunque no es definitivo.

5.2. Pruebas

5.2.1. Pruebas funcionales

Para realizar las pruebas funcionales, he desglosado los requisitos de las gestiones en 4 pruebas unitarias (consulta, inserción, edición, y borrado. He distinguido los requisitos funcionales generales de las unitarias con distinto color, pero todas tienen el mismo aspecto. Además, he añadido un requisito funcional para esta ocasión, que es el logueo para acceder a la aplicación.

RF00 LOGIN	
Precondición	Desde la pantalla de login, se pulsa en Acceder
Datos de Entrada	Se rellenan los datos del formulario
Datos de Salida	1. Si los datos de entrada son de un usuario administrador y son correctos se da paso al histórico de reservas. **OK** 2. Si los datos de entrada son del un usuario mostrador y son correctos se da paso al histórico de reservas. **OK** 3. Si los datos de entrada son del un usuario mantenimiento y son correctos se da paso a la interfaz de incidencias. **OK** 4. Si los datos son erróneos se muestra un mensaje de error. **OK**

RF01 GESTIÓN DE CLIENTES

RF01-A INSERCIÓN DE CLIENTES	
Precondición	Pulsar en el botón aceptar
Datos de Entrada	Se rellena el formulario del modal
Datos de Salida	1. Si los datos requeridos son rellenados y válidos se lleva a cabo la instrucción **OK** 2. Si los datos requeridos y rellenados no son válidos se da un mensaje de error **OK** 3. Si hay algún campo requerido vacío se muestra un mensaje de error. **OK**

RF01-B EDICIÓN DE CLIENTES	
Precondición	Hacer clic en aceptar
Datos de Entrada	Se rellena el formulario del modal
Datos de Salida	1. Si los datos requeridos son rellenados y válidos se lleva a cabo la instrucción. **OK** 2. Si los datos requeridos y rellenados no son válidos se da un mensaje de error. **OK** 3. Si hay algún campo requerido vacío se muestra un mensaje de error. **OK**

RF01-C BORRADO DE CLIENTES	
Precondición	Hacer clic en aceptar
Datos de Entrada	No hay
Datos de Salida	1. Se lleva a cabo la instrucción. **OK**

RF01-D CONSULTA DE CLIENTES	
Precondición	Acceder a la página
Datos de Entrada	No hay
Datos de Salida	1. Se visualizan los clientes en la tabla correspondiente. **OK**

RF02 GESTIÓN DE HABITACIONES

RF02-A INSERCIÓN DE HABITACIONES	
Precondición	Pulsar en el botón aceptar
Datos de Entrada	Se rellena el formulario del modal
Datos de Salida	1. Si los datos requeridos son rellenados y válidos se lleva a cabo la instrucción **OK**

	2. Si los datos requeridos y rellenados no son válidos se da un mensaje de error **OK** 3. Si hay algún campo requerido vacío se muestra un mensaje de error. **OK**

RF01-B EDICIÓN DE HABITACIONES	
Precondición	Tener abierto un modal y dar en aceptar
Datos de Entrada	Se rellena el formulario del modal
Datos de Salida	1. Si los datos requeridos son rellenados y válidos se lleva a cabo la instrucción. **OK** 2. Si los datos requeridos y rellenados no son válidos se da un mensaje de error. **OK** 3. Si hay algún campo requerido vacío se muestra un mensaje de error. **OK**

RF01-C BORRADO DE HABITACIONES	
Precondición	Hacer clic en aceptar
Datos de Entrada	No hay
Datos de Salida	1. Se lleva a cabo la instrucción. **OK**

RF01-D CONSULTA DE HABITACIONES	
Precondición	Acceder a la página
Datos de Entrada	No hay
Datos de Salida	1. Se visualizan las habitaciones en la tabla correspondiente. **OK**

RF03 GESTIÓN DE RESERVAS

RF01-A INSERCIÓN DE RESERVAS	
Precondición	Pulsar en el botón Confirmar Reserva
Datos de Entrada	Se rellenan los formularios de la página
Datos de Salida	1. Si los datos requeridos son rellenados y válidos se lleva a cabo la instrucción y se abre el modal de reserva confirmada **OK** 2. Si los datos requeridos y rellenados no son válidos se da un mensaje de error **OK** 3. Si hay algún campo requerido vacío se muestra un mensaje de error. **OK**

RF01-B EDICIÓN DE RESERVAS	
Precondición	Tener abierto el modal y dar clic en registrar check-in o registrar check-out

Datos de Entrada	
Datos de Salida	1. Se registra el check-in o el check-out y se actualiza la tabla. **OK**

RF01-C BORRADO DE RESERVAS	
Precondición	Hacer clic en aceptar
Datos de Entrada	No hay
Datos de Salida	1. Se lleva a cabo la instrucción. **OK**

RF01-D CONSULTA DE RESERVAS	
Precondición	Acceder a la página
Datos de Entrada	Se selecciona un intervalo de fechas
Datos de Salida	1. Se visualizan las reservas en la tabla correspondiente. **OK** 2. Al pulsar en el botón de detalles se abre modal con los detalles de la reserva. **OK**

RF04 FACTURAS DE ESTANCIAS

RF01-A INSERCIÓN DE FACTURAS	
Precondición	Tener abierto el modal en la pantalla de histórico de reservas y pulsar en el botón Aceptar
Datos de Entrada	Se rellena el formulario del modal.
Datos de Salida	1. Se crea la factura. **OK** 2. Si se marca como pagada, se crea la factura como pagada correctamente **OK**

RF01-B EDICIÓN DE FACTURAS	
Precondición	Tener abierto el modal en la pantalla de histórico de facturas o de reservas y pulsar en el botón Aceptar
Datos de Entrada	Se rellena el formulario del modal.
Datos de Salida	1. Se cambia la forma de pago. **OK**

RF01-C BORRADO DE FACTURAS	
Precondición	Hacer clic en aceptar
Datos de Entrada	No hay
Datos de Salida	1. Se lleva a cabo la instrucción. **OK**

RF01-D CONSULTA DE FACTURAS

Precondición	Acceder a la página
Datos de Entrada	Se selecciona un intervalo de fechas
Datos de Salida	1. Se visualizan las facturas en la tabla correspondiente. **OK**

RF07 GESTIÓN DE PLANTILLA DE PERSONAL

RF01-A INSERCIÓN DE PERSONAL	
Precondición	Pulsar en el botón aceptar
Datos de Entrada	Se rellena el formulario del modal
Datos de Salida	1. Si los datos requeridos son rellenados y válidos se lleva a cabo la instrucción **OK** 2. Si los datos requeridos y rellenados no son válidos se da un mensaje de error **OK** 3. Si hay algún campo requerido vacío se muestra un mensaje de error. **OK**

RF01-B EDICIÓN DE PERSONAL	
Precondición	Tener abierto el modal y dar en aceptar
Datos de Entrada	Se rellena el formulario del modal
Datos de Salida	1. Si los datos requeridos son rellenados y válidos se lleva a cabo la instrucción. **OK** 2. Si los datos requeridos y rellenados no son válidos se da un mensaje de error. **OK** 3. Si hay algún campo requerido vacío se muestra un mensaje de error. **OK**

RF01-C BORRADO DE PERSONAL	
Precondición	Hacer clic en aceptar
Datos de Entrada	No hay
Datos de Salida	1. Se lleva a cabo la instrucción. **OK**

RF01-D CONSULTA DE PERSONAL	
Precondición	Acceder a la página
Datos de Entrada	No hay
Datos de Salida	1. Se visualizan los empleados en la tabla correspondiente. **OK**

RF08 MANTENIMIENTO

RF08-A INSERCIÓN DE INCIDENCIAS DESDE PERSONAL MOSTRADOR	
Precondición	Pulsar en el botón aceptar
Datos de Entrada	Se rellena el formulario del modal
Datos de Salida	1. Si los datos requeridos son rellenados y válidos se lleva a cabo la instrucción **OK** 2. Si los datos requeridos y rellenados no son válidos se da un mensaje de error **OK** 3. Si hay algún campo requerido vacío se muestra un mensaje de error. **OK**

RF08-B EDICIÓN DE INCIDENCIAS DESDE PERSONAL MOSTRADOR	
Precondición	Pulsar en el botón aceptar
Datos de Entrada	Se rellena el formulario del modal
Datos de Salida	1. Si los datos requeridos son rellenados y válidos se lleva a cabo la instrucción **OK** 2. Si los datos requeridos y rellenados no son válidos se da un mensaje de error **OK** 3. Si hay algún campo requerido vacío se muestra un mensaje de error. **OK**

RF08-C BORRADO DE INCIDENCIAS DESDE PERSONAL MOSTRADOR	
Precondición	Hacer clic en aceptar
Datos de Entrada	No hay
Datos de Salida	1. Se lleva a cabo la instrucción. **OK**

RF08-D CONSULTA DE INCIDENCIAS DESDE PERSONAL MOSTRADOR	
Precondición	Acceder a la página
Datos de Entrada	Se selecciona un intervalo de fechas
Datos de Salida	1. Se visualizan las incidencias en la tabla correspondiente. **OK** 2. Al hacer clic en el botón de editar se visualiza la información completa de la incidencia. **OK**

RF08-E CONSULTA DE INCIDENCIAS DESDE PERSONAL MATENIMIENTO	
Precondición	Acceder a la página
Datos de Entrada	No hay
Datos de Salida	1. Se visualizan las incidencias pendientes en formato móvil. **OK**

RF08-F EDICIÓN DE INCIDENCIAS DESDE PERSONAL MATENIMIENTO	
Precondición	Pulsar en el botón atender incidencia, incidencia resuelta o incidencia pendiente
Datos de Entrada	Se rellena el formulario de la página
Datos de Salida	1. Se actualiza el estado de la incidencia de forma correspondiente. **OK**

RF10 GESTIÓN DE USUARIOS

RF01-A INSERCIÓN DE USUARIOS	
Precondición	Pulsar en el botón aceptar
Datos de Entrada	Se rellena el formulario del modal
Datos de Salida	1. Si los datos requeridos son rellenados y válidos se lleva a cabo la instrucción **OK** 2. Si los datos requeridos y rellenados no son válidos se da un mensaje de error **OK** 3. Si hay algún campo requerido vacío se muestra un mensaje de error. **OK**

RF01-B EDICIÓN DE USUARIOS	
Precondición	Tener abierto el modal y dar en aceptar
Datos de Entrada	Se rellena el formulario del modal
Datos de Salida	1. Si los datos requeridos son rellenados y válidos se lleva a cabo la instrucción. **OK** 2. Si los datos requeridos y rellenados no son válidos se da un mensaje de error. **OK** 3. Si hay algún campo requerido vacío se muestra un mensaje de error. **OK**

RF01-C BORRADO DE USUARIOS	
Precondición	Hacer clic en aceptar
Datos de Entrada	No hay
Datos de Salida	1. Se lleva a cabo la instrucción. **OK**

RF01-D CONSULTA DE USUARIOS	
Precondición	Acceder a la página
Datos de Entrada	No hay
Datos de Salida	1. Se visualizan los usuarios en la tabla correspondiente. **OK**

5.2.2. Pruebas de requisitos no funcionales

Los requisitos no funcionales RFN01 y RFN02 se refieren al código que no ve el cliente, por lo cual no se comprueban, ya que se comprueba en cuanto se empieza a codificar.

RFN03 ACCESIBILIDAD	
Precondición	Tener instalado varios navegadores diferentes
Datos de Entrada	Intento de entrar a la aplicación usando cada navegador
Datos de Salida	1. Si se entra con éxito en un navegador, ese navegador es aceptado. **OK**

RFN04 USABILIDAD	
Precondición	Tener distintos dispositivos
Datos de Entrada	Acceder a las interfaces desde distintos dispositivos
Datos de Salida	1. La interfaz se muestra correctamente con todos sus botones, y, según el dispositivo, cambien de tamaño **OK**

RFN05 APRENDIZAJE	
Precondición	Entrar a la aplicación
Datos de Entrada	Usar los botones
Datos de Salida	1. Los botones deben de tener iconos. **OK** 2. El menú tiene que ser claro y conciso **OK**

RFN06 SEGURIDAD DE LA RED	
Precondición	Tener instalado un servidor el local
Datos de Entrada	Trabajar en local
Datos de Salida	1. Se puede trabajar en local. **OK**

RFN07 SEGURIDAD DENTRO DE LA APLICACIÓN	
Precondición	Tener a varios usuarios y roles registrados en el sistema
Datos de Entrada	Loguearse en la aplicación
Datos de Salida	1. Si los datos de entrada son de un usuario administrador y son correctos permite visualizar todas las interfaces. **OK** 2. Si los datos de entrada son del un rol en particular, debe de acceder solo a las interfaces con permisos para ese rol. **OK**

RFN08 USUARIOS DE LA APLICACIÓN	
Precondición	Tener rol de administrador
Datos de Entrada	Acceder a la aplicación
Datos de Salida	1. El administrador debe poder agregar usuarios y asignarles un rol. **OK**

RFN09 CONCURRENCIA	
Precondición	Tener como mínimo 2 usuarios registrados
Datos de Entrada	Los usuarios se loguean al mismo tiempo, o cuando una sesión ya esté abierta
Datos de Salida	1. Se tiene que conseguir que dos o más usuarios puedan acceder a la aplicación a la vez **OK**

RF10 RENDIMIENTO	
Precondición	Estar logueado en la aplicación
Datos de Entrada	Navegar por las distintas páginas
Datos de Salida	1. La aplicación tiene que ser lo suficientemente rápida como para no impacientar al cliente. **OK**

5.3. Manual de Instalación y configuración

El objetivo de este manual es guiar al lector para configurar correctamente el equipo para que la aplicación web pueda funcionar.

5.3.1. Requisitos

- Servidor con Apache, PHP y MySQL o Mariadb como gestor de base de datos. A ser posible, con Git y Composer instalado.
- Un dominio configurado.

5.3.2. Pasos para la configuración

Estos pasos se deber hacer siguiendo el orden marcado.

1. Crear la base de datos con el programa que incorpore el servidor.
2. Clonar el repositorio. Si se dispone de Git, entonces se puede hacer con comandos usando Git:

   git clone xxxx

   Donde xxxx es la dirección del repositorio de GitHub.
3. En la carpeta del proyecto, hay un archivo llamado .env.example. Hay que abrirlo, copiar su contenido y crear otro archivo llamado simplemente .env

Configurando .env

Se edita con:

- Los datos de la base de datos (usuario y contraseña)
- La dirección del dominio
- Poner el modo producción (APP_ENV=production, APP_DEBUG=false).

El resto se puede dejar como estaba.

4. Insertar los siguientes comandos, en el siguiente orden:
    a. composer install
    b. php artisan key:generate
    c. php artisan config:cache
    d. php artisan migrate --seed

La explicación sería la siguiente:

En primer lugar, instalamos las dependencias de PHP. Después generamos una clave de encriptación de seguridad. Acto seguido cacheamos la configuración para mejorar el rendimiento de la aplicación, y finalmente ejecutamos las migraciones para crear las tablas y los seeders.

5.4. Manual de usuario

A continuación, se analiza el programa con todas sus funcionalidades con detalle para que el lector se familiarice con la herramienta.

5.4.1. Login

Al acceder a la aplicación, lo primero que el usuario ve es la pantalla de Login o logueo. Se le presentan 2 campos:

 En este campo se pone el nick de usuario de quien se vaya a loguear.

 En este campo se escribe la contraseña de la cuenta con la que se vaya a loguearse. Si se desea, se da la opción de mostrar la contraseña haciendo clic en el símbolo del ojo.

Por último se le da en el botón Aceptar o la tecla Enter para acceder.

Fig 5.6 Página de login de la aplicación web

Los usuarios con permisos de administrador y mostrador acceden directamente a la página de reservas, que se verá más adelante.

Dentro de la aplicación se mostrará una barra superior con el nombre de usuario en la esquina superior derecha, donde podemos hacer clic en cualquier momento para cerrar la sesión.

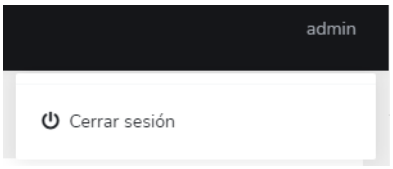


Fig 5.7 Menú sencillo de sesión con el botón Cerrar sesión.

5.4.2. Menú o sidebar

Una vez dentro de la aplicación los usuarios con permisos de administrador y mostrador se encuentran con un menú con las distintas páginas a su izquierda. “Habitaciones” es un desplegable que contiene a su vez dos links a dos páginas, habitaciones y Tipos de habitaciones.

Dependiendo de los permisos, el menú tendrá enlaces a todas las páginas o a unas cuantas.

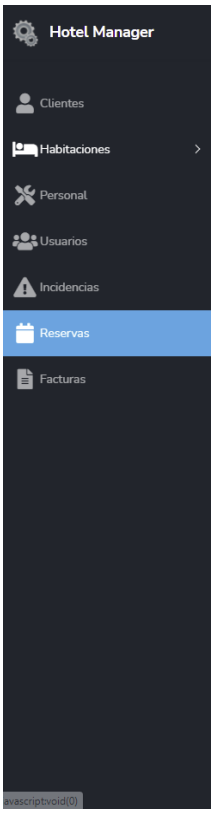


Fig. 5.8 El menú como lo ve el administrador. Puede acceder a todas las páginas del menú

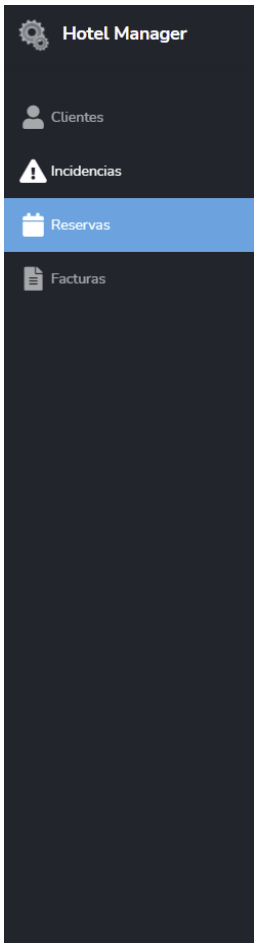


Fig. 5.9 El menú como lo ven los usuarios con el permiso de mostrador.

Se comprueba que, en este caso, los que tienen el permiso de mostrador sólo pueden acceder a la página de clientes, de incidencias, de reservas y de facturas, mientras que el administrador puede acceder a todas las páginas.

Hay páginas que no tienen enlace en el menú que se describirán más adelante.

5.4.3. Páginas de las que se compone la aplicación

5.4.3.1. Clientes (visible para el administrador y los usuarios con el permiso “Mostrador”)

Al dar clic en el enlace Clientes del menú, se abre la página de clientes, donde se muestra una tabla con los datos personales de los clientes registrados en la base de datos.

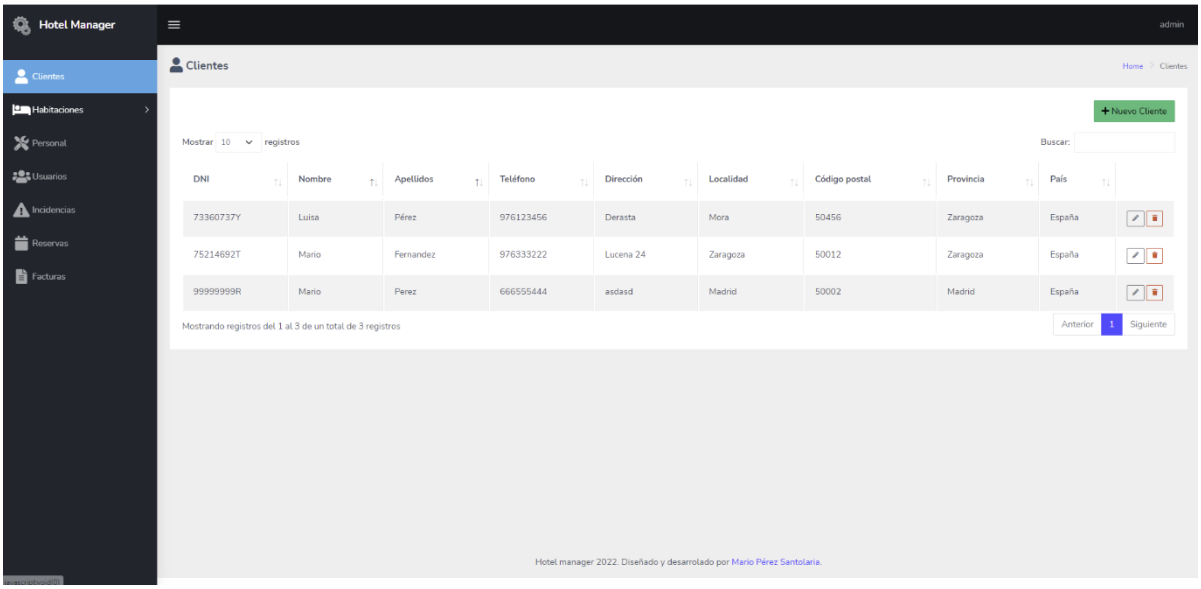


Fig. 5.10 Página de clientes

Registrar un nuevo cliente

Para añadir un nuevo cliente, hacemos clic en el botón “Nuevo Cliente”, situado en la esquina superior derecha de la página, encima de la tabla.

Nos sale una ventana que tendremos que rellenar con los datos personales del cliente que se vaya a registrar. Rellenamos los datos y hacemos clic en aceptar.

En el caso de que el cliente ya exista, saldrá un mensaje de error en el campo al dar clic.

Fig. 5.11 Ventana Nuevo cliente

El cliente queda registrado y se podrán ver sus datos en la tabla.

La última columna de cada fila está reservada para las acciones que queramos hacer. Los botones que se encuentran son el de editar ( ) y el de borrar ( ).

Editar un cliente

El boton editar nos muestra la misma ventana de Nuevo cliente pero con los datos rellenados. Podemos cambiar cualquier campo y le hacemos clic en Aceptar. Los datos del cliente se han actualizado.

Borrar un cliente

Cuando hacemos clic sobre el botón borrar nos sale un mensaje de advertencia preguntándonos si queremos borrarlo.

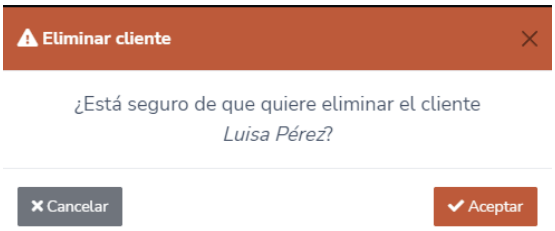


Fig 5.12 Ventana Eliminar cliente.

Si queremos borrar al cliente le damos clic en Aceptar.

Nota: Si ya existe una reserva asociada a ese cliente, el cliente no se puede borrar.

5.4.3.2. Tipos de habitaciones (visible para el administrador)

El enlace a esta página se sitúa en el submenú “Habitaciones” dentro del menú principal.

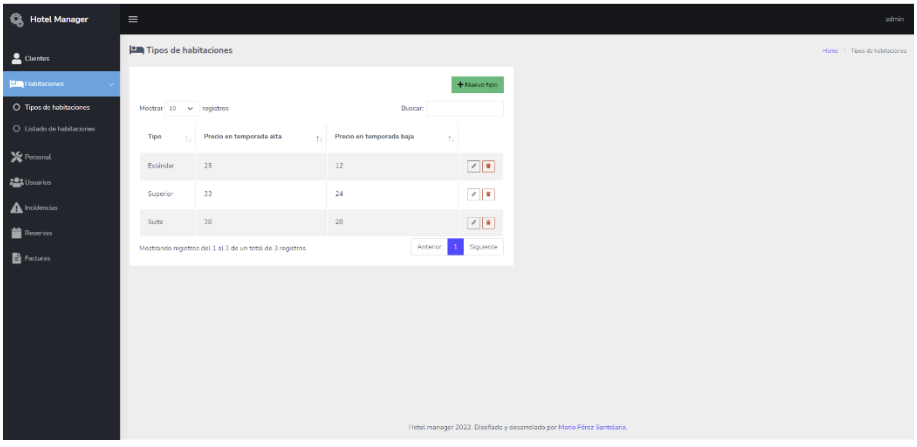
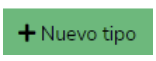


Fig 5.13 Página de tipos de habitaciones, dentro del submenú Habitaciones

Crear un tipo de habitación



Para crear un nuevo tipo de habitación, pulsaremos sobre el botón que dice “Nuevo tipo”. Se nos abrirá una ventana, donde se nos pide el nombre de tipo de habitación y sus precios, tanto en temporada alta como en temporada baja.

Fig 5.14 Vetana Nuevo tipo de habitación

Al darle clic en Aceptar se registra el tipo de habitación y los datos son mostrados en la lista de los tipos de habitaciones.

Editar un tipo de habitación

Si queremos editar un tipo de habitación, pincharemos en el botón editar en la fila correspondiente ( ) Se abre la ventana "Editar tipo", similar a la ventana "Nuevo tipo", con los datos que tiene el tipo de habitación seleccionado. Al darle a Aceptar, los cambios se guardan, y podremos ver en la tabla los datos actualizados.

Borrar un tipo de habitación

Si queremos borrar un tipo de habitación, le daremos al botón de la papelera ( ). Se nos abrirá una ventana preguntando si queremos borrar el tipo. Hacemos clic en Aceptar.

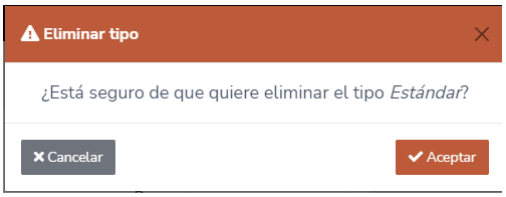


Fig 5.15 Ventana "Eliminar tipo de habitación"

El tipo de habitación se borra.

Nota: Si ya existe una habitación del tipo que se quiere borrar, el tipo no se podrá borrar.

5.4.3.3. Listado de habitaciones (visible para el administrador)

Se accede a través del submenú "Habitaciones", en el menú principal. Aquí se muestran todas las habitaciones, cuántas personas pueden alojarse en cada habitación, los precios de cada una y si están disponibles para la reserva.

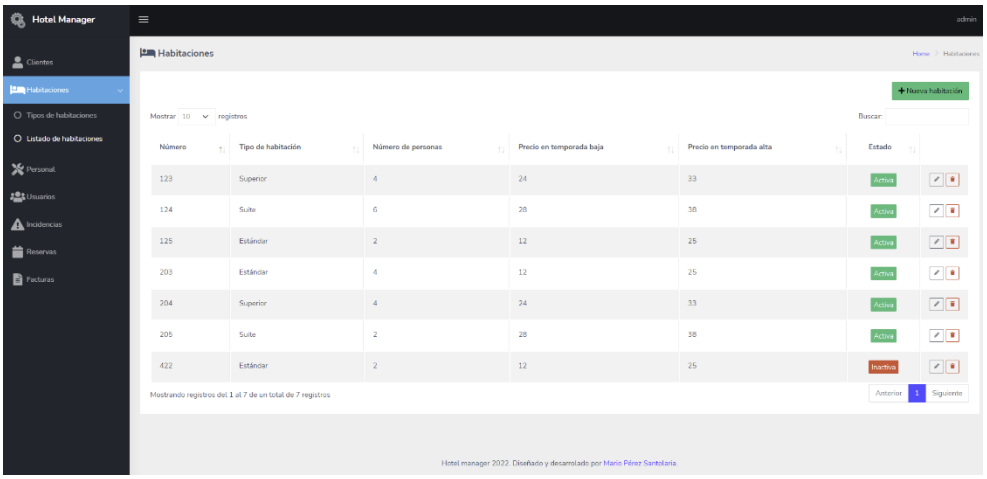
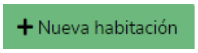


Fig 5.16 Página de habitaciones

Añadir una nueva habitación



Para añadir una nueva habitación, haremos clic en “Nueva habitación”. Nos sale una ventana para añadir los datos de la habitación.

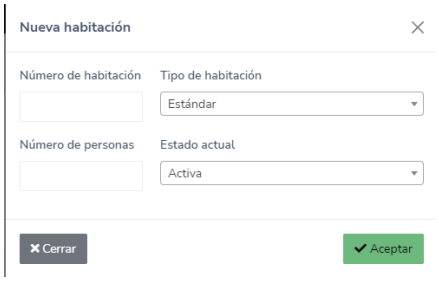


Fig 5.17 Ventana “Nueva habitación”

Añadimos el número, el tipo de habitación que es (estándar, suite…) dependiendo de los tipos de habitación que haya, el número de personas y su estado actual. Para añadir la habitación le damos al botón Aceptar.

Editar una habitación

Para editar una habitación pulsamos el botón de edición (). No saldrá la ventana Editar habitación, similar a la de nueva habitación con los datos actuales. Rellenamos los campos que queremos editar y, al darle en el botón Aceptar, se actulizarán los datos y se podrán ver en la tabla.

Borrar una habitación

Para borrar una habitación, tenemos que pulsar el botón con el icono de la papelera () y saldrá un mensaje preguntanso si queremos borrar la habitación. Aceptamos y la habitación se borrará.

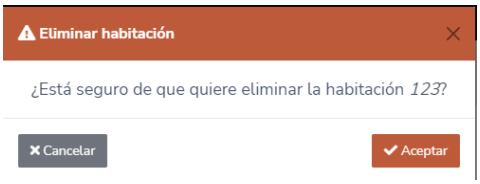


Fig 5.18 Ventana “Eliminar habitación”

Nota: Si ya existe una reserva o una incidencia asociada a esa habitación, no se puede borrar la habitación.

5.4.3.4. Personal (visible para el administrador)

La página “Personal”, que se accede mediante el menú, muestra el personal registrado. Esto incluye tanto los encargados del mantenimiento como los recepcionistas. La tabla muestra los datos personales de cada uno, así como al tipo que pertenece.

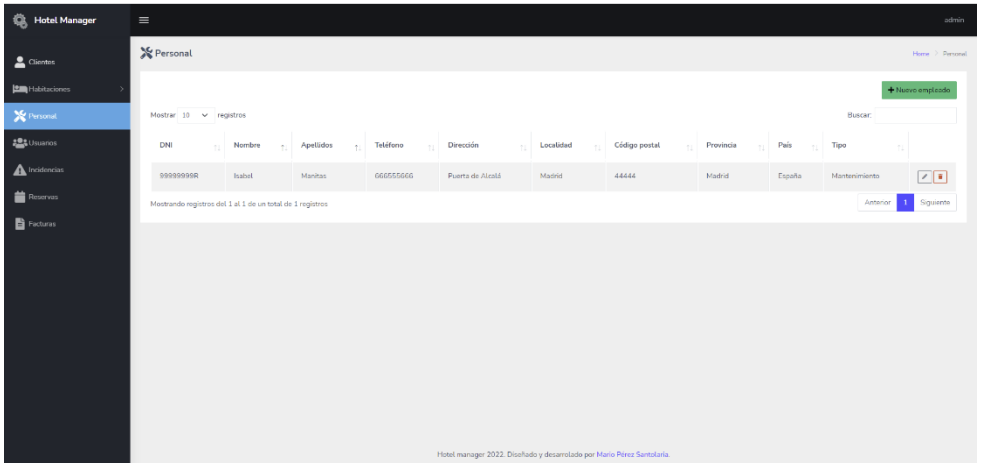


Fig 5.17 Página “Personal”

Añadir un empleado

Para añadir un nuevo empleado hacemos clic en el botón “Nuevo empleado”. Nos sale la ventana de nuevo empleado.

Fig 5.18 Ventana “Nuevo empleado”

Rellenamos los datos y en tipo ponemos el tipo de personal, si es de mantenimiento (permiso de mantenimiento) o un recepcionista (permiso de mostrador).

Editar un empleado

Para editar un empleado, pulsaremos en el botón editar(). Nos sale una ventana similar a la de Nuevo empleado, con los datos ya puestos. Cambiamos los que queremos actualizar, y le damos clic en Aceptar. Los datos del empleado se actualizan.

Borrar un empleado

Para borrar un empleado, pulsaremos sobre el botón eliminar (). No sale la dvertencia del borrado del empleado. Le damos clic en Aceptar.

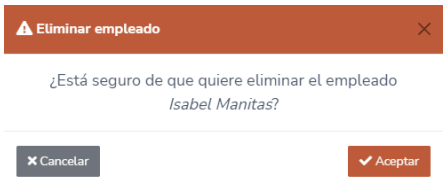


Fig. 5.19 Ventana “Eliminar empleado”.

5.4.3.5. Usuarios (visible para el administrador)

Se accede a través del menú. Muestra los distintos usuarios registrados en la aplicación hasta el momento: usuario y contraseña. Además, muestra el perfil de cada usuario (cada perfil tiene unos permisos diferentes) y el empleado asociado, refiriéndose a su nombre y apellidos.

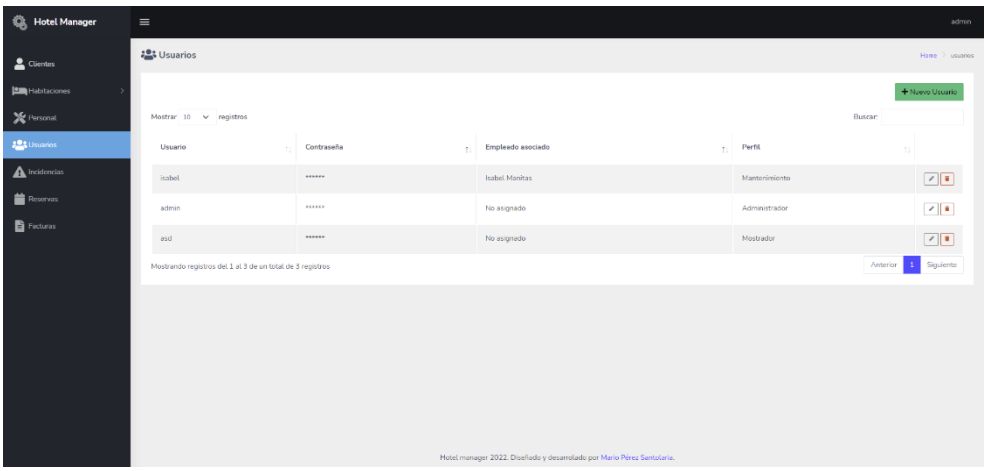


Fig 5.20 Página “Usuarios”

Añadir un usuario

Para añadir un nuevo usuario a la aplicación, daremos clic en Nuevo usuario.

Nos sale la ventana de nuevo usuario.

Fig 5.21 Ventana “Nuevo usuario”

Ingresamos el nuevo nick de usuario y la contraseña. En empleado asociado podemos elegir quién del personal existente es el que a a usar la cuenta. Por último, seleccionamos el perfil que tendrá, y le damos en Aceptar.

Editar un usuario

El botón editar ( ) nos permite editr los datos de una cuenta. Se nos abre una ventana similar a la de nuevo usuario con los campos ya rellenados. Una vez hayamos cambiado el dato que queramos cambiar, damos clic en Aceptar.

Nota: Aunque no se haya editado la contraseña, a la hora de cambiar algun dato hay que ponerla nuevamente, aunque siga siendo la misma.

Eliminar un usuario

Para eliminar un usuario tenemos que hacer clic en el botón eliminar ( ). Saldrá una ventana indicando que se va a borrar el usuario. Le damos clic en Aceptar.

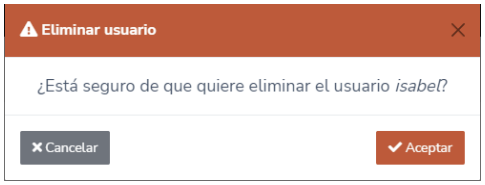


Fig 5.21 Ventana “Eliminar usuario”

5.4.3.6. Incidencias (Visible para el administrador y los usuarios con el permiso de mostrador)

Cuando pinchamos en el enlace Incidencias del menú, se muestra la lista de incidencias que hay ordenadas por defecto por la fecha de notificación que tienen. Se muestran los siguientes datos:

- Urgencia. Hay 3 tipos de urgencia:
  - Urgente. Incidencia que ha de resolverse en cuanto se pueda.
  - Moderada. Urgencia más leve que debe ser resuelta pero no inmediatamente.
  - No urgente. Pequeña urgencia que debe ser resuelta, pero se puede dejar para más adelante.
- Habitación. El número de la habitación que tiene la incidencia.
- Descripción. Una pequeña descripción de la incidencia que el administrador puede añadir cuando se crea la incidencia.
- Fecha de notificación. Fecha y hora de la notificación de la incidencia.
- Atendida. La persona que atendió la incidencia. Puede que no haya sido atendida por nadie aún, en ese caso se mostrará una etiqueta que dice “Sin atender”.
- Fecha de resolución. Fecha y hora que fue resuelta la incidencia.

Crear una incidencia

Puede crear una nueva incidencia haciendo clic en + Nueva Incidencia. Se abre la ventana “Nueva incidencia” que permite seleccionar el tipo de urgencia, la habitación, ingresar una breve descripción (la que deespués se muestra en la tabla) y escribir detalles (este campo es obligatorio también). Hay un apartado para las acciones, pero el administrador no puede escribir nada, puesto que lo hará la persona que resuelva la incidencia.

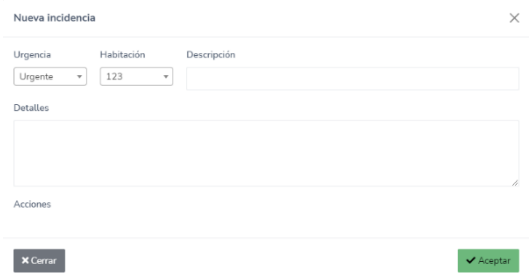


Fig 5.22 Ventana “Nueva incidencia”

Editar una incidencia

Se puede editar una incidencia dándole clic en el botón editar (). Se abre una ventana similar a la de nueva incidencia. Si el encargado ha rellenado el apartado Acciones, saldrá aquí. Si no, permanecerá vacío, sin dejar que se rellene. Editamos los campos que queramos cambiar y le damos en Aceptar.

Borrar una incidencia

Para borrar una incidencia tenemos que hacer clic en el botón de eliminar ().Se muestra la ventana "Eliminar incidencia", advirtiendo del borrado de la incidencia. Le damos clic en Aceptar.

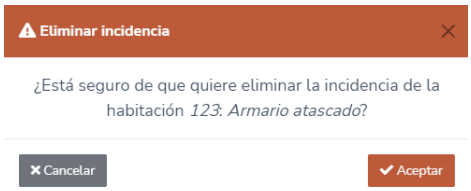
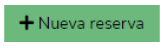


Fig 5.23 Ventana "Eliminar incidencia"

5.4.3.7. Reservas y facturas (visibles para el administrador y los usuarios con el permiso de mostrador)

Se acceden a través del menú.

Crear una reserva



Para crear una reserva presionamos en el botón de nueva reserva, que nos lleva a una página distinta, nueva reserva.

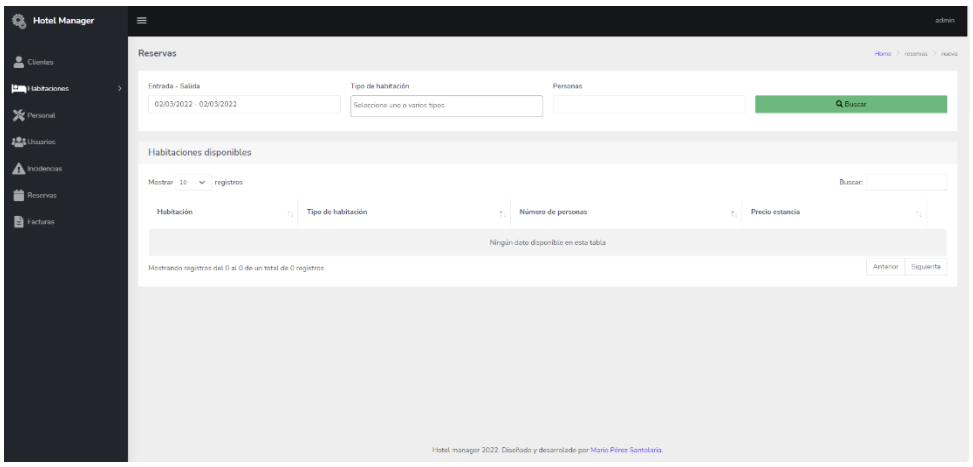


Fig 5.24 Página "Nueva reserva". Esta página solo se puede acceder desde la página Reservas.

Nos encontramos dos partes diferenciadas:

- Un formulario donde podemos elegir el periodo de la estancia que nos indique el cliente que vaya a alojarse. El tipo de habitación es una selección (puede ser múltiple en caso de que al cliente no le importe demasiado el tipo) y el número de personas que hayan solicitado quedarse. Acto seguido, damos clic en Buscar.
- Una tabla que es el resultado de la búsqueda anterior. Inicialmente está vacía. Esta tabla indica los datos de las habitaciones disponibles que cumplen las condiciones

puestas en el formulario. En consecuencia, cualquiera de los resultados servirá para la reserva. Dependiendo de la ocupación, puede que no se muestre ninguna. En ese caso, los criterios deberán ser otros, y la reserva tendrá otros datos.

Cuando haya una o más habitaciones disponibles, le damos al botón seleccionar (Seleccionar) de la habitación elegida, cuya fila se resaltará.

Al darle al botón Seleccionar, en la parte inferior aparecerá un nuevo espacio marcado como “Cliente”.

- El espacio “Cliente” es otro formulario en el que debemos poner el DNI del cliente. El sistema lo buscará. Si está registrado, se ve a continuación sus datos para asegurarse de que los datos se corresponden. Si no existe, se le puede registrar pulsando “Nuevo cliente”, que abre la ventana descrita en la página Clientes.

  Se le da clic en Confirmar Reserva para reservar la estancia. Ahora se muestran los datos de la reserva en la tabla de la página “Reservas”.

Fig 5.25 Espacio “Clientes” con los datos del cliente encontrado

Fig 5.26 Notificación de reserva confirmada

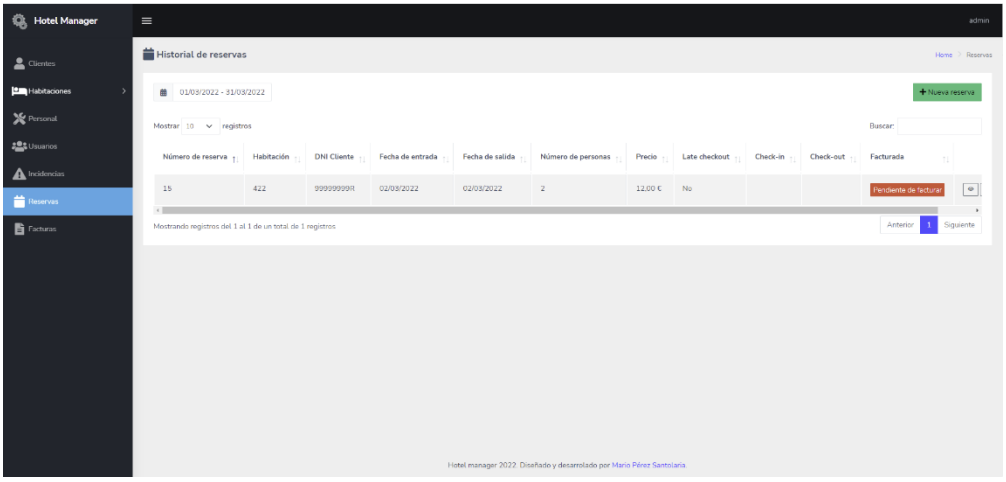


Fig 5.27 Página de reservas con la reserva añadida.

Observamos que la reserva queda pendiente de facturar. También vemos cómo se han

rellenado todos los datos (el número de reserva, el número de la habitación, la fecha de entrada y de salida, el precio, y si tiene Late check-out o no) menos el check-in y el check-out.

Editar una reserva

Si nos fijamos, hay 3 botones en la última columna de cada reserva.

- El botón del ojo ( ) nos permite ver de una forma más detallada la reserva separando por un lado la información de la reserva y por el otro la información del cliente que realizó la reserva.

  Se nos presenta la posibilidad de registrar el check-in o el check-out, dependiendo de la situación. Si ya se ha registrado el check-out no se verá ningún botón.

Fig 5.28 Detalles de la reserva.

Si se registra el check-in se verá la fecha y hora actual en dicha columna. Lo mismo ocurrirá cuando le demos al check-out. El check-out solo estará disponible si anteriormente se ha hecho el check-in.

- El botón facturar ( ) abre una nueva ventana, “Facturar reserva”. Se muestran algunos datos de la reserva.
  - Si queremos facturar la reserva, le daremos al botón Aceptar. El estado pasará a ser “Pendiente de pago”, y la factura la podremos ver en “Facturas”.
  - Si la factura está generada, y el cliente la ha pagado marcaremos la casilla “Pagada”, y seleccionamos en el desplegable la forma de pago de la reserva. Aceptamos, el estado pasará a ser “pagada”, y la factura la podremos ver en “Facturas”.
  - Si le damos a cerrar, la ventana se cerrará, y el estado no cambiará.

Si se ha facturado una reserva, independientemente de su estado, se podrá ver detalladamente en la ventana.

Fig 5.29 Ventana "Facturar reserva"

Borrar una reserva

El botón borrar ( ) abre una ventana con el mensaje de borrado de una reserva. Le damos a Aceptar.

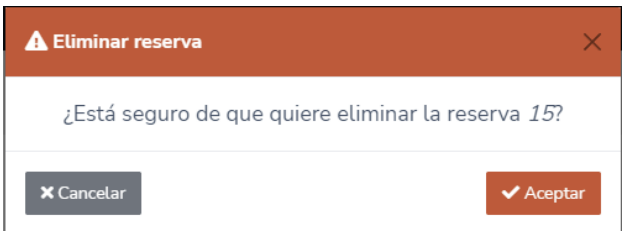


Fig 5.30 Ventana "Eliminar reserva"

Nota: La reserva que ya esté facturada no se puede borrar. Se ha de eliminar la factura para poder borrar la reserva.

La página de facturas es una lista de las facturas generadas. El botón editar ( ) cumple la misma función que el botón editar de las reservas. El botón borrar ( ) permite borrar una factura. Este paso es necesario hacerlo para borrar la correspondiente reserva.

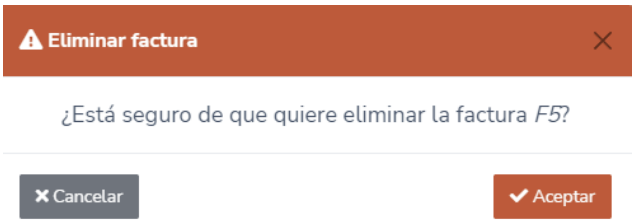


Fig 5.31 Ventana "Eliminar factura"

5.4.3.8. Lista de incidencias (mantenimiento) (visible para el administrador y usuarios con el permiso "Mantenimiento"

Se accede cuando se loguea con un usuario con permisos de mantenimiento. Se le redirige a esta página, optimizada para un dispositivo móvil, que lista las incidencias que se han marcado en la página "Incidencias" por el administrador. Cada incidencia muestra el número de habitación, fecha y hora de la notificación y una breve descripción de la incidencia, además del tipo de urgencia que tiene.

Fig. 5.32 Lista de incidencias vista por el personal de mantenimiento

Cuando se le da clic en una incidencia se muestra los detalles, un botón de atender incidencia, los detalles proporcionados por el administrador, y que el personal de mantenimiento puede editar, y el apartado de acciones, donde el personal de mantenimiento podrá editar, al contrario que el administrador.

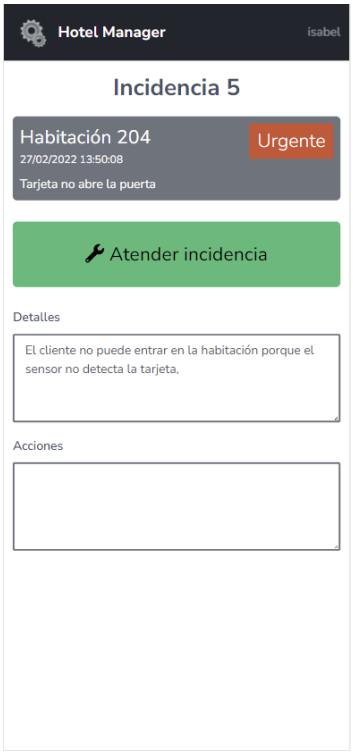


Fig 5.33 Página de detalles de la incidencia.

Cuando alguien le da al botón de atender incidencia, se rellena el campo Atendido en la páginas Incidencias del administrador con el nombre de la persona que accede a atender la incidencia. Se muestra la siguiente pantalla:

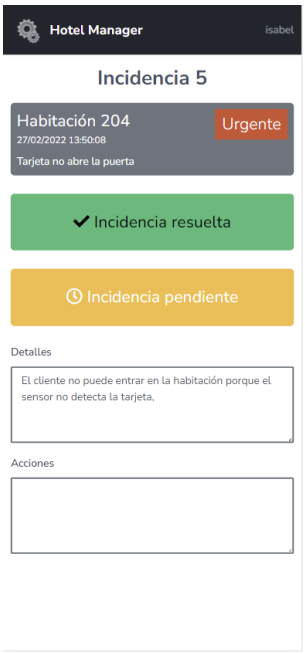


Fig 5.34 Una vez confirmada la petición de atender la incidencia, se muestra esta pantalla

Puede ocurrir dos cosas:

- Si se le da a Incidencia resuelta, la incidencia se resuelve. Por tanto, desaparece de la lista, y en la página de Incidencias se rellena el campo Fecha de Resolución con la fecha y hora actuales.
- Si se le da a Incidencia pendiente, la incidencia queda pendiente, pero sigue asignada a la persona. En la lista visible para el mantenimiento seguirá, y en la pantalla Incidencias, aunque el nombre de la persona se refleje, aún no habrá ninguna fecha de resolución, por lo que ese campo seguirá vacío.

El campo Acciones de los detalles que ve el administrador se corresponde con el campo que edita la persona de mantenimiento.

Si hay alguna incidencia que aún no ha sido atendida, se verá una etiqueta de “Sin atender” en lugar del nombre de la persona en la tabla que ve el administrador.

DOCUMENTO DE CIERRE

6. Documento de cierre

6.1. Resultados obtenidos y conclusiones

6.1.1. Resultados obtenidos

No me veo del todo satisfecho con el resultado, más que nada porque considero que mi aplicación es bastante básica. Creo que he cubierto todas las necesidades esenciales que pueda tener y de eso sí que me quedo satisfecho, aunque no he podido completar otras funcionalidades no esenciales que quería hacer por falta de tiempo, como implementar la funcionalidad de imprimir las facturas.

La mayor dificultad que me tropezado, o una de las más grandes, es la utilización de un framework tan complejo como Laravel. He tenido que comprender la estructura, cómo funcionaba el MVC y, sobre todo, repasarme la sintaxis de JavaScript y aprender cómo funciona AJAX. Sin embargo, gracias a que anteriormente tenía conocimientos de programación y a base de cometer muchos errores, cuando supe cómo funcionaba y sobre todo el flujo de los datos, pude elaborar una aplicación web funcional.

Aún así, el cliente ha quedado contento con el resultado.

6.1.2. Conclusiones

Aunque todo el proyecto no ha salido como en un principio se esperaba, el trabajo me ha permitido darme una idea de cómo pueden ser los proyectos en la vida laboral de un programador. Me ha permitido ver que todo proyecto necesita su documentación y he comprendido el objetivo de ésta. También he madurado haciendo un proyecto real, que sea aceptable por clientes verdaderos. Me ha permitido estudiar nuevos conceptos, aunque solo sea de forma práctica (aparte del framework y todos los plugins, no conocía el tema de despliegue), y poder aplicarlos. Me anima a hacer otros programas como trabajador y como hobby.

Sin embargo, no todo son luces, y no me planteo repetir esta experiencia en el futuro a no ser que sea de forma obligada, es decir, si hago una aplicación hacia un cliente ficticio o para mí, la documentación que genere sería de otra forma, con dibujos que comprenda mejor, que me ayuden a pensar los procedimientos que debo hacer en cada momento, y de cada página y menos detallada y, por supuesto, sin tanto estilo como la que nos ocupa. O directamente sin documentación.

6.2. Diario de bitácora

Durante todo el proyecto he usado Git y he generado varios commits que, a pesar de que no ha sido de manera periódica sino irregular, he terminado acostumbrándome a hacerlas de forma casi continua. He hecho un total de 27 commits, de los cuales 17 están en la rama máster, en internet. Por supuesto, el más reciente es de cómo ha quedado la aplicación. El enlace del repositorio en GitHub es el siguiente.

https://www.github.com/Mariops89/projectHotelManager

BIBLIOGRAFÍA

7. Bibliografía

WEBS

1. Gestion ágil vs Gestión tradicional de proyectos ¿Cómo elegir?, de Escuela de Negocios FEDA. (www.escueladenegociosfeda.com/blog/50-la-huella-de-nuestros-docentes/471-gestion-agil-vs-gestion-tradicional-de-proyectos-como-elegir). Consultado el 8 de noviembre de 2021
2. Font Awesome icons. (www.fontawesome.com/icons) Visitado por última vez el 22 de febrero de 2022.
3. Web oficial de Moment. (www.momentjs.com/). Consultado por última vez el 22 de febrero del 2022.
4. Web oficial de Select 2. (www.select2.org/). Consultado por última vez el 22 de febrero del 2022.
5. Web oficial de Datatables. (www.datatables.net/). Consultado por última vez el 22 de febrero del 2022.
6. Web oficial de Daterangepicker. (www.daterangepicker.com/) Consultado por última vez el 22 de febrero del 2022.
7. AJAX. Guía de Desarrollo Web – MDN (www.developer.mozilla.org/es/docs/Web/Guide/AJAX). Consultado el 22 de febrero de 2022.
8. JavaScript – MDN (www.developer.mozilla.org/es/docs/Web/JavaScript). Consultado el 22 de febero de 2022.
9. HTML: Lenguaje de Marcas de Hipertexto (www.developer.mozilla.org/es/docs/Web/HTML). Consultado el 22 de febero de 2022.
10. CSS – MDN (www.developer.mozilla.org/es/docs/Web/CSS) Consultado el 22 de febrero de 2022.
11. DOM – Glosario – MDN (www.developer.mozilla.org/es/docs/Glossary/DOM) Consultado el 22 de febrero de 2022.
12. Jquery – Glosario – MDN (www.developer.mozilla.org/es/docs/Glossary/jQuery). Consultado el 22 de febrero de 2022.
13. Bootstrap: ¿qué es, para qué sirve y cómo instalarlo? (www.rockcontent.com/es/blog/bootstrap) Consultado el 21 de febrero de 2022.
14. Web oficial de Bootstrap. Consultado por última vez el 08 de febrero de 2022.
15. Web oficial de Laravel. Consultado por última vez el 08 de febrero de 2022.
16. Introducción a las migraciones con Laravel Framework. (www.anexsoft.com/introduccion-a-las-migraciones-con-laravel-framework) Consultado por última vez el 20 de enero de 2022.
17. Generador de números NIF, NIE, CIF. (www.testingdatagenerator.com/doi.html). Consultado por última vez el 20 de febrero de 2022.
18. Specificity Calculator. (www.specificity.keegan.st/) Consultado el 07 de febrero de 2022)
19. Despliegue de aplicaciones Web con Deployer (I). (www.conasa.grupocibernos.com/blog/despliegue-de-aplicaciones-php-con-deployer-i) Consultado el 20 de febrero de 2022.

20. Laravel-spanish-validator. (www.github.com/orumad/laravel-spanish-validator) Consultado el 27 de diciembre de 2021.
21. XAMPP vs WAMP. ¿Cuál es el mejor servidor local para desarrollo web? (www.todoxampp.com/xampp-vs-wamp-cual-es-el-mejor-servidor-local-para-desarrollo-web). Consultado el 22 de diciembre de 2021.
22. ¿Qué es JavaScript? Características y librerías. (www.miteris.com/blog/que-es-javascript-caracteristicas-librerias/). Consultado el 22 de diciembre de 2021.
23. Características de phpMyAdmin – Tutoriales de SiteGround. (www.siteground.es/tutoriales/tutoriales-phpmyadmin/caracteristicas/) Consultado el 22 de diciembre de 2021.
24. Qué es Jquery y sus características. (www.openwebinars.net/blog/que-es-jquery/) Consultado el 22 de diciembre de 2021.
25. ¿Qué es Laravel? (www.arsys.es/blog/que-es-laravel) Consultado el 29 de noviembre de 2021.
26. Ventajas y desventajas de PHP (www.baulphp.com/ventajas-y-desventajas-del-lenguaje-php/). Consultado el 29 de noviembre de 2021.
27. Documentación oficial de Bootstrap 5. (www.getbootstrap.com/docs/5.1/getting-started/introduction/). Consultado por última vez el 08 de febrero de 2022.
28. Web oficial de Laravel 9. (www.laravel.com/docs/9.x). Consultado por última vez el 08 de febrero de 2022.

CURSOS

29. Despliegue de aplicaciones web, por Sergio Pacheco. Curso 2021/2022 - CFPIFP Los Enlaces.

ANEXOS

8. Anexos

8.1. Presupuesto completo

Tipo	Descripción	Cantidad	Precio	Subtotal
Material	Equipo (Hardware + Software)	1 unidad	420 € / unidad	420
Licencia	Licencia de PHPStorm	3 meses	18 € / mes	54
Servicio	Luz	3 horas	70 € / hora	210
Servicio	Teléfono + Internet	3 horas	70 € / hora	210
Rol	Analista	39 horas	30 € / hora	1170
Rol	Diseñador	32 horas	20 € / hora	640
Rol	Programador	84 horas	15 € / hora	1260
Rol	Probador	16 horas	10 € / hora	160
Rol	Documentación	12 horas	10 € / hora	120
Otro	Alquiler sala reuniones	6 horas	11 € / hora	66
	TOTAL			**4310 €**

8.2. Contrato

CONTRATO DE DISEÑO WEB

En Zaragoza, a 28 de febrero de 2022

REUNIDOS

D. Mario Pérez Santolaria, mayor de edad, con DNI 72994689A y domicilio en Zaragoza, actuando en nombre y representación de "Proyectos del Mundo" S. L. inscrita en el Registro Mercantil de Zaragoza con domicilio social en C/ Bilbao, 208, actuando en su calidad de programador, en posesión de poderes suficientes para este acto. (El Desarrollador)

D Pablo Santos Santos, mayor de edad, con DNI 65437650Z y domicilio en Palma de Mallorca, actuando en nombre y representación de "Hotel Mar", inscrita en el Registro Mercantil de Palma de Mallorca en calidad de administrador, en posesión de poderes suficientes para este acto. (El cliente)

MANIFIESTAN

Que las partes están interesadas en formalizar el presente contrato; que poseen suficientes poderes para la firma del mismo; que se reconocen capacidad legal necesaria para poder llevar a cabo la celebración y declaran expresamente que actúan de forma libre, voluntaria y no viciada.

EXPONEN

El cliente está interesado en que el prestador lleve a cabo el diseño y desarrollo de una página Web conforme a las necesidades especificas indicadas en el Anexo I.

Que el prestador tiene como objeto el diseño y desarrollo de páginas y aplicaciones Web y cuenta con el personal necesario para ello; que las características de la página Web son las indicadas en el Anexo I por el cliente; que ambas partes aceptan cumplir con sus respectivas obligaciones.

En relación con lo anteriormente expuesto, las partes otorgan el presente contrato que se regirá por las siguientes

CLAUSULAS

I. OBJETO DEL CONTRATO

La aplicación Web se acogerá en todo caso a las categorías, diseño y contenidos indicados en el Anexo I.

II. PROPIEDAD INTELECTUAL

El prestador garantiza al cliente que todo el material utilizado para el desarrollo del proyecto, así como el resultado obtenido, es un producto original que no vulnera ninguna ley o derechos de terceros, en especial los referidos a propiedad industrial e intelectual.

El prestador reconoce los derechos de explotación sobre la obra. El prestador renuncia de forma indefinida a ejercitar cualquier tipo de acción para recuperar sus derechos de propiedad intelectual respecto al desarrollo, a excepción del derecho de autoría por el que el desarrollador tiene derecho a exigir que aparezca su nombre o un link al su sitio Web.

En caso de ser el cliente el encargado de proporcionar los contenidos (gráficos, textos, vídeos, categorías…), éste se hace responsable de cualquier tipo de reclamación de terceros en relación a la titularidad de dichos contenidos, eximiendo de toda responsabilidad al prestador.

III. OBLIGACIONES DEL PRESTADOR

El prestador se compromete a desarrollar el presente proyecto bajo las directrices del cliente, ajustándose a los términos indicados por éste en el Anexo I y conforme a las mejores prácticas existentes en el mercado, así como con la máxima diligencia posible.

Una vez aceptadas las características del sitio Web pueden producirse variaciones en el diseño y/o contenido de este a petición del cliente. Salvo que conllevaran una variación sustancial del Proyecto Inicial no supondrá aumento del precio, considerando variación sustancial toda aquello que suponga un aumento del tiempo total de trabajo estimado inicialmente superior al 10 %.

El prestador se compromete a finalizar el desarrollo en el plazo acordado siempre que la otra parte haya colaborado activamente en el desarrollo de este (aceptando los diseños, entregando los contenidos, etc.)

IV. OBLIGACIONES DEL CLIENTE

El cliente se obliga a realizar el pago del precio en los términos indicados en la cláusula VI del presente contrato.

El cliente se obliga a mantener un contacto constante con el prestador entregando en tiempo y forma los contenidos del proyecto Web (Textos, imágenes, videos, categorías…), la aceptación del diseño y cualquier otra necesidad que requiera el prestador para poder finalizar el proyecto. En cualquier caso, se atenderá a lo dispuesto respecto a propiedad intelectual en la cláusula II.

En caso de depender la entrega de contenidos de un tercero seleccionado por el cliente, éste deberá indicarlo en el apartado comunicaciones y comprometiéndose a responder de los posibles retrasos que pudieran darse.

V. COMUNICACIONES

Las partes se obligan a comunicarse toda la información que pudiera ser necesaria para el correcto desarrollo del proyecto. Toda comunicación entre las partes relativa al presente contrato se realizará por escrito o telefónicamente. A efectos de comunicaciones y/o notificaciones las partes designan:

Prestador

Domicilio en Zaragoza, calle Bilbao 208, con número de Fax 976353535, correo electrónico oficina@proyectosmundo.com y Teléfono 666666000

Cliente

Domicilio en Palma de Mallorca, calle del Rosal, 11, con número de fax, 945454545 correo electrónico oficina@hotelmar.com y Teléfono 945464646

Cualquier cambio de domicilio o dirección de contacto deberá ser comunicado a la otra parte por escrito con una antelación mínima de10 días hábiles.

VI. DURACIÓN Y PRECIO

El presente contrato entrará en vigor el día 19de octubre de 2021 y tendrá una duración de 6 meses, siendo posible, a petición del prestador, añadir 10 días adicionales para llevar a cabo la entrega del proyecto, sin ello suponer repercusión económica alguna.

El precio a abonar por parte del cliente como pago por la prestación del servicio prestado equivale a 4310 € (cuatro mil trescientos diez euros), añadiendo a dicha cantidad el IVA correspondiente (16%)

Dicho precio será abonado de la siguiente forma:

a) 1000 € (mil euros), que serán abonados en el momento de la firma del presente contrato.

b) La cantidad restante, equivalente a 3310 € (tres mil trescientos diez euros), serán abonados en el momento de finalización del proyecto.

El pago final será llevado a cabo siempre que el prestador de por finalizado el proyecto y en cualquier caso cuando se cumpla el plazo de entrega y el cliente no haya entregado o colaborado activamente en la entrega de contenidos, validación de colores y diseño o entrega de categorías del sitio Web.

El pago será realizado mediante transferencia Bancaria en el número de cuenta ES01 0101010101019999999.

VIII. CONFIDENCIALIDAD Y PROTECCIÓN DE DATOS

Las partes contratantes declaran conocer y cumplir la legislación europea y española sobre protección de datos, comprometiéndose a tratar los datos personales obtenidos durante el desarrollo del proyecto de acuerdo con dicha normativa.

Se informa al cliente que sus datos quedarán almacenados en un fichero, automatizado o no, con las únicas finalidades de informarle sobre novedades y nuevos proyectos en los que se encuentre trabajando la empresa prestadora, así como para el mantenimiento de la relación contractual.

Se informa al afectado que puede ejercer los derechos de acceso, rectificación, cancelación y oposición de sus datos de carácter personal solicitándolo por escrito y acompañando una fotocopia del DNI en la dirección del prestador indicada en el presente contrato.

Toda la información relativa al proyecto, así como a los datos de carácter personal, tendrá el carácter de confidencial por lo que las partes deberán guardar el mayor secreto respecto a las mismas, garantizando en todo caso el no acceso por parte de terceros a dicha información.

XI. NO COMPETENCIA

El cliente se compromete a no realizar proyectos de equivalente o análoga naturaleza para un tercero, ni iniciar a partir del presente desarrollo una nueva línea de negocio relacionada con la realización de sitios Web. Se considerará que el cliente incurre en dicha circunstancia siempre que opere directa o indirectamente mediante otra empresa en la que disponga de participación social, o que actúe como mero asesor o colaborador y que en definitiva obtenga como resultado un proyecto igual, semejante o con la misma finalidad a la ofrecida por el prestador.

El incumplimiento del anterior compromiso llevará aparejada una penalización equivalente a 1000 €, sin perjuicio y de las indemnizaciones que correspondiesen por los daños y perjuicios causados al prestador.

XI. EXTINCIÓN

Además de por las causas generales del Derecho, este contrato se extinguirá:

a. Por el transcurso del mismo.

b. Por ser declarados en situación de suspensión de pagos, quiebra o concurso de acreedores cualquiera de las partes.

c. Por incumplimiento de las obligaciones estipuladas en el presente contrato.

Las partes aceptan expresamente el sometimiento a las penalizaciones económicas indicadas a continuación siempre que se rescinda el contrato de forma previa a la finalización del proyecto:

XII JURISDICCIÓN Y LEGISLACIÓN APLICABLE

Todas las cuestiones litigiosas sobre el presente contrato mercantil quedarán regidas por la legislación española, específicamente por lo dispuesto en el Código Mercantil, y en su defecto, por las disposiciones españolas del Código de Comercio, Leyes Especiales, usos mercantiles y con carácter supletorio, por el Código Civil.

En cualquier caso, será obligatorio que en caso de conflicto las partes intenten previamente resolver la cuestión de mutuo acuerdo, sometiéndose en su caso a los Juzgados y Tribunales de Zaragoza que por orden correspondan.

ANEXO I: PROYECTO TÉCNICO

La aplicación consiste en un programa de gestión hostelera, encargada de las gestionar facturas, reservas de los clientes, registro de clientes, listados de habitaciones y control de labores de mantenimiento. La idea es encontrar un método de gestión más eficiente y rápido que el de antes.

- **Características del Diseño.**

  La aplicación web es un conjunto de páginas con un menú lateral que lleva a las diferentes páginas. Cada página se muestra una tabla que contiene la información que se requiere.

- **Colores Corporativos.**

  Los colores corporativos son el verde oliva y el negro o gris.

- **Imágenes y logotipos.**

  Las imágenes e iconos son elegidos por el programador.

- **Formato de entrega de los contenidos.**

  Vendrá en soporte USB el contenido necesario para instalar la aplicación en el servidor.

  También se le entregarán dos copias de la documentación: una en formato digital con extensión PDF incluida en el USB, y otra impresa en papel.

- **Planificación de las tareas.**

  Adjunto una imagen con las tareas previstas, así como las fechas. La fecha de fin prevista es para el **18 de diciembre del 2021** si no hay imprevistos de última hora.

Nombre	Fecha de inicio	Fecha de fin
Investigación del mercado actual y posible solución. Objetivos del proyecto.	19/10/21	25/10/21
Descripción del proyecto	1/11/21	5/11/21
Documento de acuerdo del proyecto	25/10/21	18/11/21
Requisitos	25/10/21	11/11/21
Tareas a realizar	11/11/21	12/11/21
Metodología y elección de un modelo de ciclo de vida	12/11/21	12/11/21
Análisis de posibles riesgos y errores que puedan suceder. Creación análisis DAFO	16/11/21	16/11/21
Planificación temporal	18/11/21	18/11/21
Presupuesto (gastos, ingresos y beneficio)	18/11/21	18/11/21
Contrato y condiciones de ambas partes	18/11/21	18/11/21
Primer Sprint	19/11/21	2/12/21
Análisis y diseño del primer sprint.	19/11/21	22/11/21
Diseño del sistema y arquitectura de software: realización de diagramas.	22/11/21	25/11/21
Realización de mockups.	26/11/21	26/11/21
Reunión con el cliente.	26/11/21	26/11/21
Desarrollo del core del sistema: montar bbdd, plantilla html y funcionamiento básico del backend.	26/11/21	26/11/21
Desarrollo de frontend según los mockups e integración dentro del core del sistema	26/11/21	2/12/21
Pruebas de software del primer sprint	2/12/21	2/12/21
Segundo Sprint	2/12/21	14/12/21
Reunión con el cliente.	2/12/21	2/12/21
Análisis y diseño del segundo sprint.	2/12/21	3/12/21
Cambios en el software del sprint 1	3/12/21	7/12/21
Implementación del backend del proyecto	7/12/21	10/12/21
Interconexión backend - frontend	13/12/21	14/12/21
Pruebas de software del segundo sprint	14/12/21	14/12/21
Pruebas completas de la aplicación web	14/12/21	15/12/21
Implementación final: últimos retoques	13/12/21	14/12/21
Pruebas completas de la aplicación web	14/12/21	15/12/21
Documentación del proyecto	15/12/21	17/12/21