Grado en Ingeniería Informática

Trabajo Final de Grado

Gestor de reservas de casas

Autor:
Pau Alegria Civera

Supervisor:
Antonio Vera
Tutor académico:
Dolores María Llidó Escrivá

Fecha de lectura: 17 de Julio de 2017
Curso académico 2016/2017

Resumen

Este documento corresponde al trabajo de fin de grado (TFG) de la asignatura "Prácticas externas y proyecto de fin de grado EI1054 2016/2017" del grado de informática. En él se detalla de forma técnica el proceso de elaboración de un proyecto en el ámbito de las tecnologías Web en el periodo de estancia en prácticas en la empresa Angal Informatica. El proyecto propuesto por la empresa es una aplicación web de gestión de casas rurales y sus reservas.

Para realizar el proyecto se ha optado por una metodología tradicional, realizando una planificación, análisis, diseño, implementación y pruebas desde un enfoque de la Ingeniería del Software.

Para la implementación de la aplicación web se ha usado principalmente PHP como lenguaje del lado del servidor, JavaScript como lenguaje del lado del cliente, Bootstrap como complemento para el diseño de interfaces y MySQL como gestor de base de datos.

Palabras clave

Desarrollo web, PHP, HTML, CSS, JavasScript, MySQL, UML, Diagramas de casos de uso, TFG, Ingeniería Informática.

Keywords

Web development, PHP, HTML, CSS, JavasScript, MySQL, UML, Use case diagrams, Final Project Computer Engineering Degree.

Índice general

Capítulo 1

Introducción

Este capítulo es una breve descripción general del TFG, así como la de empresa, el proyecto propuesto, sus objetivos y los del alumno. En el último punto se describe cómo está estructurada la memoria.

1.1. Contexto y motivación del proyecto

Esta memoria es el resultado del trabajo del alumno tras cursar la asignatura "Prácticas externas y proyecto de fin de grado EI1054 2016/2017". En ella se detalla el desarrollo de un proyecto propuesto por una empresa en el transcurso de la estancia en prácticas.

La empresa seleccionada es Angal Informática, una empresa relativamente joven, ubicada en Castellón de la Plana que ofrece soluciones informáticas en el ámbito del software, hardware e internet. La empresa está especializada en ofrecer herramientas y servicios e-comerce, páginas web personalizadas, marketing, redes sociales y soporte de sistemas informáticos.

El proyecto desarrollado, consiste en una plataforma web para gestionar reservas de casas rurales. Para ponernos en situación, Angal Informática, tiene diferentes clientes dedicados al sector de alquileres de casas rurales, los cuales, gestionan sus reservas de una forma tradicional (bolígrafo, papel y teléfono). Estas empresas de alquiler de casas, a su vez, ya tienen sus propios portales web de promoción desde los cuales sus clientes pueden ver las casas que ofertan. Lo que se desea, es ampliar dichos portales para que permitan que cualquier persona pueda realizar reservas directamente a través de ellos.

Desde la perspectiva del alumno, se ve este proyecto como una oportunidad de desarrollar un producto comercial desde cero a partir de una necesidad real de una empresa. Además de adquirir experiencia en entornos web y lenguajes altamente conocidos y demandados como son el PHP y el JavaScript.

1.2. Objetivos del proyecto

Podemos destacar distintos niveles de objetivos según el punto de vista del alumno, de la empresa y del proyecto propiamente dicho.

Desde el punto de vista del estudiante tiene como objetivos:

- Elaborar un proyecto de naturaleza profesional en el ámbito de las tecnologías.
- Mostrar su capacidad en el uso de las metodologías y las herramientas de la Ingeniería del Software.

Desde el punto de vista de la empresa tiene como objetivos:

- Generar una nueva línea de negocio.
- Crear una nueva necesidad a sus clientes.
- Mejorar la funcionalidad y la utilidad de sus actuales webs dedicadas a la promoción de casas rurales.

Desde el punto de vista del proyecto tiene como objetivos:

- Cubrir los requisitos funcionales y de datos del proyecto.
- Mejorar la actual metodología de trabajo de administradores de casas rurales.
- Ser una herramienta útil, fácil de usar y atractiva para los administradores de casas rurales y sus posibles clientes.

Los objetivos planificados no se han llegado a ejecutar en su totalidad debido a que la empresa no tenía bien definidos los requisitos y como consecuencia ha afectado a la finalización de la parte pública de la aplicación.

1.3. Estructura de la memoria

En el Capítulo 2 se presenta una descripción detallada del proyecto. El Capítulo 3 muestra la metodología, planificación y seguimiento que se ha realizado del proyecto. En el Capítulo 4 se detalla el análisis. En el Capítulo 5 el diseño y el desarrollo donde se incluyen pruebas y por último en el Capítulo 6 las conclusiones.

Capítulo 2

El proyecto

En las siguientes secciones se describe con detalle el proyecto con sus objetivos y cuáles han sido las tecnologías usadas para su desarrollo.

2.1. Descripción del proyecto

Como ya se ha comentado en puntos anteriores, el proyecto a desarrollar consiste en la creación de una plataforma web para gestionar reservas de casas en alquiler. La empresa de prácticas Angal Informática, tiene diferentes clientes dedicados al sector de alquileres de casas rurales, los cuales, gestionan sus reservas de una forma tradicional (bolígrafo y papel). Una metodología de trabajo basada en el uso del teléfono para contactar con los usuarios y una agenda de reservas donde realizan sus anotaciones a mano. Estas empresas de alquiler de casas, a su vez, ya tienen sus propios portales web para su promoción. Lo que desea la empresa es ampliar dichos portales para que permitan realizar reservas directamente a través de ellos. Este proyecto se centrará en realizar una aplicación de gestión de las reservas informatizada suficientemente genérica para que se adecúe a las necesidades de cualquier cliente, es decir, cualquier empresa del sector de alquiler de casas. Hay que tener en cuenta que el alcance de este proyecto no abarca la integración del gestor de reservas que vamos a desarrollar con ninguna otra web. Dicho de otra forma, la aplicación de reservas se desarrollará como un ente autónomo. Posteriormente Angal Informática se encargará de la integración de la aplicación con los sistemas existentes de sus clientes.

Para ayudar a esclarecer la funcionalidad del proyecto vamos a empezar describiendo los distintos tipos de usuarios que interactúan con él:

- Los usuarios no autentificados, también los podríamos denominar como clientes del gestor de reservas, son aquellas personas que tienen interés en disfrutar de alguna de las casas de alquiler que ofrece la plataforma. Para ello, la aplicación de gestión ha de permitir que éstos usuarios puedan buscar de entre las distintas casas ofertadas la que más se adecúe a sus necesidades. Posteriormente, cuando el usuario decida qué casa disfrutar, el gestor ha de permitirle realizar la reserva de la misma.

- Los usuarios autentificados, son aquellos encargados de administrar los datos de la aplicación de gestión. Estos usuarios han de identificarse en la plataforma, los cuales, una vez validados se les dará acceso a su panel de gestión. Las tareas que han de realizar estos usuarios gestores son variadas, las más destacadas son las relacionadas con las reservas, las casas y las tarifas o precios de las temporadas. En los puntos siguientes haremos una descripción más detallada de las funcionalidades que permite el gestor de reservas a los usuarios autentificados.

Una vez descritos los usuarios, vamos a describir las funcionalidades que ha de brindar la aplicación de gestión de reservas según sus vistas. Entendemos como vista, las interfaces o pantallas que se encargan de interactuar con los usuarios. Hay dos tipos de vistas, la vista Frontend relacionada con el usuario no autentificado, la cual la podemos especificar como pública para cualquier persona y la vista Backend o gestión interna, relacionada con el usuario autentificado, la cual es privada.

Funcionalidades que debe permitir la vista Frontend:

- Se debe implementar un buscador donde los no autentificados puedan realizar las reservas de las casas. En este apartado la plataforma deberá mostrar la información de todas las casas que se adapten a las necesidades del cliente.
- El gestor permitirá que el usuario seleccione una fecha de entrada y salida y el número de personas, además de otra serie de criterios opcionales como número de habitaciones, de camas o las características de la casa así como si tiene actividades relacionadas. Posteriormente se mostrará como resultado una lista de casas que se adecúen a las condiciones seleccionadas por el cliente. Cada casa listada incluirá sus precios además de permitir ser reservada por el cliente.
- En la aplicación de gestión se tendrá que registrar la información de la reserva incluyendo aquellos datos proporcionados por el usuario no autentificado como son los datos personales, los de la casa seleccionada y las fechas de la reserva.
- El Frontend ha de permitir un sistema de autentificación, donde el usuario del gestor especificará el nombre de usuario y password para acceder a la gestión interna.

Funcionalidades que debe permitir la vista interna o Backend:

- Que los usuarios autentificados de forma sencilla puedan ver qué reservas hay y a su vez puedan gestionarlas. Podrán aceptarlas, modificarlas o rechazarlas.
- Por otro lado debe permitir crear nuevas reservas por si algún cliente prefiere comunicarse por teléfono como lo hacen actualmente.
- También se han de administrar las casas susceptibles de reservas con aquellas características interesantes para la reserva así como sus tarifas asociadas o periodos de inactividad.
- El gestor debe permitir diversos usuarios gestores y además algún tipo de sistema de validación de los mismos.

- También debe incluirse algún tipo de sistema que se encargue de llevar un registro de las acciones de los usuarios gestores.
- Por último, la plataforma incluirá interfaz de seguimiento, que mediante una serie de gráficos y estadísticas pueda mostrar al gestor ver el estado de ocupación de su negocio basándose en los datos de las reservas.

Ahora se describe la información que ha de manejar la plataforma:

- Para la reservas el gestor mantendrá la información del cliente que ha realizado la reserva, como el email, el nombre completo, el número de personas, las fechas de entrada y salida, alguna información adicional que se desee incluir y la casa seleccionada para su disfrute, así como el precio total y las tarifas incluidas.
- De cada casa la aplicación de gestión deberá permitir mantener información sobre la ubicación del edificio (población y provincia), número de habitaciones, número de camas, número de baños, número máximo de personas, nombre descriptivo, tarifas base y de fin de semana, posibles descuentos, características adicionales, posibles actividades lúdicas que se puedan realizar y posibles periodos de cierre con su descripción ya sea por mantenimiento o por temporada baja.
- Para cada casa se debe poder asociar diferentes tarifas según las temporadas, como por ejemplo para la temporada de Semana Santa a las casas se les puede asignar unas tarifas especiales distintas de sus tarifas base o de fin de semana.

2.2. Objetivos del proyecto

El principal objetivo que ha establecido la empresa Angal Informática, es realizar una plataforma web que mejore la actual comunicación con los clientes y la metodología de trabajo de aquellas empresas del sector de la reserva de casas.

Para ello se han destacado dos objetivos que la plataforma web ha de cumplir. El primer objetivo consiste en un Frontend para que los clientes de la plataforma, los usuarios no autentificados, puedan examinar las casas disponibles y a su vez reservarlas de forma sencilla y online. Y el segundo objetivo consiste en desarrollar un Backend para gestionar de forma sencilla y eficiente las reservas de las casas.

Para ello, la plataforma desarrollada debe permitir los siguientes puntos:

- Dos vistas diferenciadas, una privada, Backend, encargada de gestionar la información y otra pública, Frontend, encargada de ofertar las casas.
- Un sistema de seguridad, que garantice el no libre acceso a los datos privados del gestor.
- Hay información privada a la que los usuarios no autentificados no deben tener acceso.

- El Frontend debe facilitar a los usuarios no autentificados un modo de reserva online, fuera de horarios de atención al cliente y sin la necesidad de intermediarios.
- Informar a los usuarios no autentificados de forma rápida y sencilla de qué casas hay disponibles en todo momento y el precio de las mismas. Esto mejora la forma de informar a los usuarios de la casas disponibles.
- El usuario autentificado ha de gestionar de forma eficaz las casas que posteriormente se ofrecerán. Aquí se debe tener un control sobre las características, los precios y las fechas de cierre de las casas.
- El Backend ha de permitir una gestión eficiente del calendario de las reservas, con sus fechas, su duración y la información del cliente. Lo que se desea es que no se solapen reservas y ayudar al usuario autentificado en su labor diaria.
- El usuario autenticado ha de poder gestionar las reservas aceptándolas, modificarlas o rechazarlas según el criterio del usuario gestor.
- El Backend ha de poder mantener precios especiales según las temporadas.
- El usuario autentificado con un simple vistazo a un interfaz debería poder ver si la plataforma está generando sus expectativas comerciales.
- Llevar un control de las acciones y accesos en el Backend. Esto permitirá contralar posibles acciones no deseadas y poder corregir errores no deseados.

2.3. Tecnologías del proyecto

Desde un primer momento la empresa de estancia en prácticas, ha tenido claro qué tecnologías querían para el proyecto. Son tecnologías con las que están familiarizados y fuertemente integradas en sus proyectos.

De parte del cliente se han usado las tecnologías relacionadas con el browser como son los lenguajes estándares HTML, CSS y JavaScript. Bootstrap para diseño de la web con su enfoque de móviles primero y jQuery como librería de JavaScript para ayuda en la programación de parte del cliente. Además de un tema o interfaz llamado ADMLITE 2 como multitud de ejemplos de interfaces para administración, el cual está desarrollado en Bootstrap 3, jQuery y con componentes de terceros.

HTML y CSS. [1] HTML (Hypertext Markup Language) y CSS (Cascading Style Sheets) son dos de las principales tecnologías para crear páginas web. HTML proporciona la estructura de la página, CSS la disposición (visual y auditiva), para una variedad de dispositivos. Junto con gráficos y secuencias de comandos, HTML y CSS son la base de la construcción de páginas web y aplicaciones web.

JavaScript. [2] JavaScript® (a veces abreviado como JS) es un lenguaje ligero e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web. El estándar de JavaScript es ECMAScript. Desde el 2012, todos los navegadores

modernos soportan completamente ECMAScript 5.1. Los navegadores más antiguos soportan por lo menos ECMAScript 3.

jQuery. [3] JQuery es una biblioteca JavaScript rápida, pequeña y rica en funciones. Hace cosas como el desplazamiento y manipulación de documentos HTML, manejo de eventos, animación y Ajax mucho más simple con una API fácil de usar que funciona a través de una multitud de navegadores. Con una combinación de versatilidad y extensibilidad, jQuery ha cambiado la forma en que millones de personas escriben JavaScript.

Bootstrap 3. [4] Un elegante, intuitivo y potente framework basado en el concepto de "mobile first" con el fin de facilitar y agilizar el desarrollo web.

De parte del servidor, se propuso apache como servidor HTTP, MySQL como gestor de bases de datos y PHP como lenguaje de programación.

Apache. [5] Apache HTTP Server project es un esfuerzo para desarrollar y mantener un servidor HTTP de código abierto para sistemas operativos modernos incluyendo UNIX y Windows. El objetivo de este proyecto es proporcionar un servidor seguro, eficiente y extensible que proporcione servicios HTTP en sincronización con los estándares HTTP actuales.

MySQL Server. [6] Gestor de base de datos de código abierto.

PHP.[7] PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

Nos parece importante comentar que el alumno propuso hacer uso de algún framework para PHP, pero se desestimó dicha propuesta, por considerar que la curvatura de aprendizaje de un framework implicaría un retraso en la elaboración del proyecto.

Capítulo 3

Planificación del proyecto

A continuación se describe la metodología usada para el proyecto 3.1 y su planificación asociada 3.2. Posteriormente haremos una estimación del presupuesto del proyecto 3.3 y finalmente cómo se ha hecho el seguimiento del proyecto 3.4.

3.1. Metodología

Angal informática, la empresa de prácticas, suele trabajar desarrollando proyectos pequeños, del tipo portales web y tiendas online donde apenas se plantean usar una metodología desde el punto de vista de la Ingeniería del Software. Usan un gestor de tareas para sus proyectos y van realizando dichas tareas según su criterio de prioridad.

En un principio el proyecto propuesto por la empresa es pequeño, de una estimación de 300 horas, donde el cliente es la misma empresa y los requisitos iniciales y las tecnologías están definidos. Partiendo de estos criterios el alumno se ha decantado por un modelo en cascada, que es una metodología predictiva donde las fases del proyecto son secuenciales y sólo requiere una revisión al final de cada fase, además de ser un modelo sencillo y simple de usar.

Teniendo en cuenta el proyecto y adecuándolo a las características de la asignatura se han definido las siguientes fases y subfases más representativas:

- Desarrollo de la propuesta técnica.
    - Definición del proyecto.
    - Alcance y objetivos.
    - Planificación.
- Desarrollo técnico del proyecto.
    - Análisis.
    - Diseño.
    - Implementación y pruebas.
- Documentación y entrega del proyecto.

3.2. Planificación

La estimación temporal del TFG es un total de 450 horas que corresponden al total de horas de la asignatura EI1054 - Prácticas Externas y Proyecto de Final de Grado, donde se estipula que hay que realizar 316 horas presenciales y 134 no presenciales. Para el proyecto se ha planificado 300 horas correspondientes a las prácticas externas en la empresa y 150 horas que se dividen en la realización de la propuesta técnica, informes semanales, la memoria del proyecto, las reuniones con el tutor y por último la presentación del proyecto. Para la realización de la planificación se ha hecho uso de la herramienta Microsoft Project.

Como se ha comentado en el punto anterior, se han definido una serie de fases que se realizarán de forma secuencial. En cada una de las fases y subfases se han definido varias tareas. Tal y como puede verse en la tabla de la Figura 1, también se han definido las dependencias entre ellas, así como una estimación de la duración en horas de cada una de ellas. Además es posible ver el día de inicio y el día de finalización de cada tarea. La planificación temporal también se ve reflejada en el Diagrama de Gantt, que se puede ver gráficamente en la imagen de la Figura 2.

Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras	Nombres de los recursos
1		**Reservas de casas rurales**	**672 horas**	**mar 20/12/16**	**vie 14/04/17**		
2		**Desarrollar la propuesta técnica**	**168 horas**	**mar 20/12/16**	**mar 17/01/17**		
3		**Inicio**					
4		Definición proyecto	3 horas	mar 20/12/16	mar 20/12/16		Jefe proyecto
5		Definir métodos de trabajo y documentación	5 horas	mar 20/12/16	mié 21/12/16	4	Jefe proyecto
6		**Documentar y planificar el proyecto**					
7		Revisar contexto y buscar información	5 horas	mié 21/12/16	jue 22/12/16	5	Analista 2
8		Identificar el alcance y objetivos	5 horas	jue 22/12/16	vie 23/12/16	7	Jefe proyecto
9		**Planificar el proyecto**					
10		Definir tareas y fechas	3 horas	vie 23/12/16	lun 26/12/16	8	Jefe proyecto
11		Crear el diagrama de Gantt	5 horas	lun 26/12/16	mar 27/12/16	10	Jefe proyecto
12		Documentar la propuesta del proyecto	10 horas	mar 27/12/16	jue 29/12/16	11	Jefe proyecto
13		Entregar la Propuesta Técnica	5 horas	jue 29/12/16	vie 30/12/16	12	Jefe proyecto
14		Revisión del tutor de la propuesta técnica	2 horas	vie 30/12/16	vie 30/12/16	13	Jefe proyecto
15		Modificación y actualización de la propuesta técnica	5 horas	vie 30/12/16	lun 02/01/17	14	
16		Entrega de propuesta técnica revisada	1 hora	lun 02/01/17	lun 02/01/17	15	Jefe proyecto
17		**Desarrollo técnico del proyecto**	**574,33 horas**	**lun 02/01/17**	**mié 12/04/17**		
18		**Definir los requisitos de proyecto**	**16 horas**	**lun 02/01/17**	**mié 04/01/17**		
19		Crear diagrama de Casos de Uso	5 horas	lun 02/01/17	mar 03/01/17	16	Analista 2
20		Definir y documentar los requisitos de datos	5 horas	mar 03/01/17	mié 04/01/17	19	Analista 1
21		Definir los requisitos tecnológicos y de plataforma	5 horas	lun 02/01/17	mar 03/01/17	16	Analista 1
22		**Análisis**	**37,67 horas**	**mié 04/01/17**	**mié 11/01/17**		
23		Crear diagrama de Clases	5 horas	mié 04/01/17	jue 05/01/17	20	Analista 2
24		Documentar clases	5 horas	jue 05/01/17	vie 06/01/17	23	Analista 1
25		Diagrama de actividades	10 horas	vie 06/01/17	mar 10/01/17	24	Analista 1
26		Validar el análisis	3 horas	mar 10/01/17	mié 11/01/17	25	Analista 1
27		**Diseño**	**55,67 horas**	**mié 11/01/17**	**vie 20/01/17**		
28		Identificar y clasificar los usuarios	5 horas	mié 11/01/17	jue 12/01/17	26	Analista 2
29		Diseñar las interfaces gráficas	25 horas	mié 11/01/17	mié 18/01/17	26;21	Analista 1
30		Refinar el diagrama de clases de diseño	5 horas	mié 18/01/17	jue 19/01/17	29	Analista 1
31		Validar el diseño	5 horas	jue 19/01/17	vie 20/01/17	30	Analista 1
32		**Desarrollar el producto**	**465,67 horas**	**vie 20/01/17**	**mié 12/04/17**		
33		Creación de interfaces	9 horas	vie 20/01/17	lun 23/01/17		Programador
34		Programación	233 horas	mar 24/01/17	mié 29/03/17		Programador
35		Pruebas	40 horas	vie 31/03/17	mié 12/04/17	34	Programador
36		Entrega final	1 hora	mié 12/04/17	mié 12/04/17	35	Programador
37		**Documentación y presentación del proyecto**	**585,33 horas**	**mar 03/01/17**	**vie 14/04/17**		
38		Redacción de informes quincenales					Jefe proyecto
39		Redacción de la memoria técnica	90 horas	mar 03/01/17	vie 27/01/17	16	Jefe proyecto
40		Entrega de la memoria técnica	1 hora	mié 12/04/17	mié 12/04/17	39;36	Jefe proyecto
41		Preparación de la presentación oral	8 horas	mié 12/04/17	vie 14/04/17	40	Jefe proyecto
42		Presentación oral	1 hora	vie 14/04/17	vie 14/04/17	41	Jefe proyecto

Figura 1: Tareas

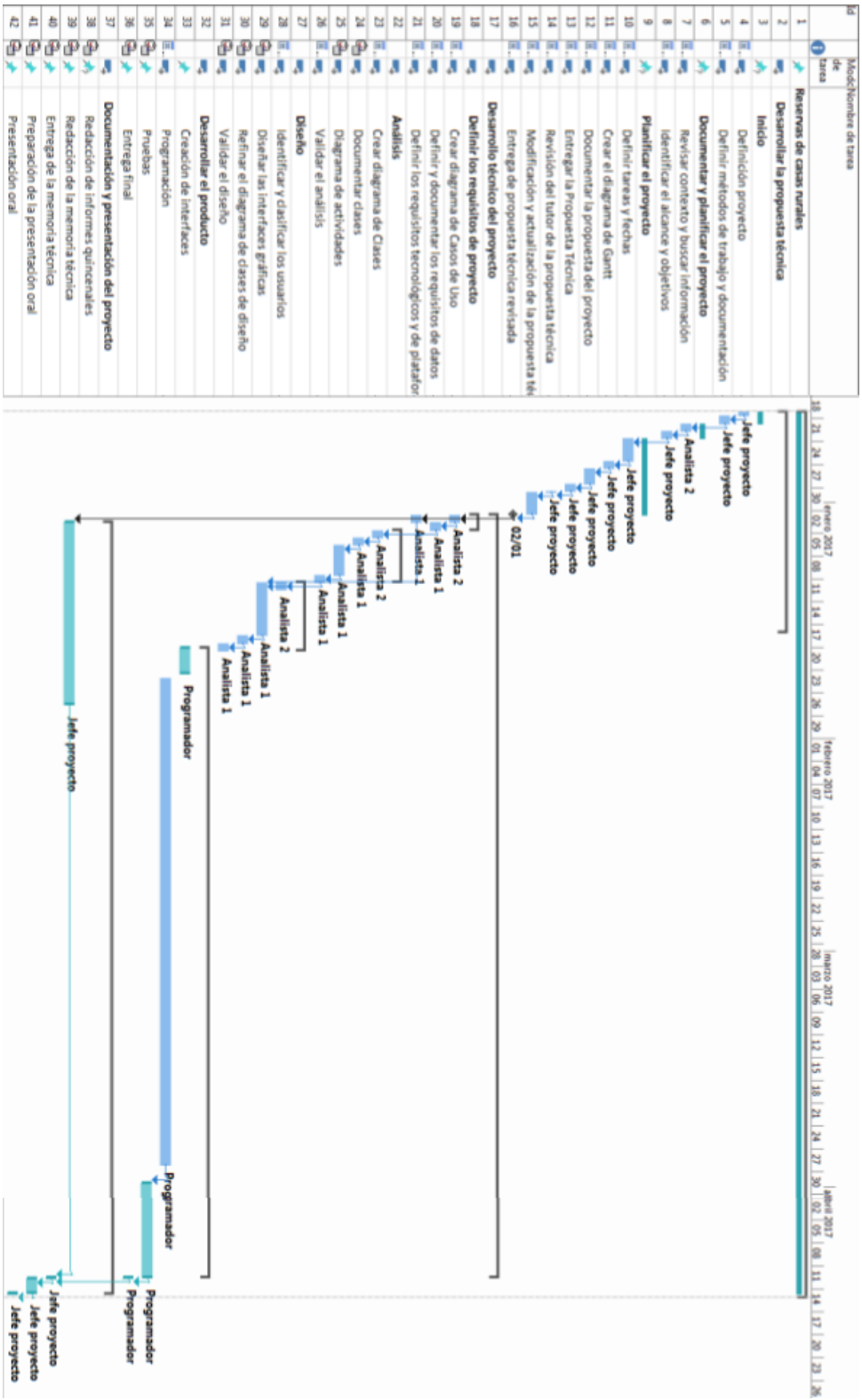

Figura 2: Diagrama de Gantt

3.3. Estimación de recursos y costes del proyecto

Se han valorado los siguientes recursos para el proyecto:

- 2 Analistas.
- 1 Programador.
- 1 Jefe de proyecto.

Se ha decidido tener dos analistas, uno para las tareas de análisis y diseño en la empresa de prácticas y otro dedicado a las tareas de documentación relacionadas con el análisis y diseño para elaboración de la memoria. Un programador para las tareas de implementación y pruebas del proyecto. El jefe de proyecto como responsable de la propuesta técnica, reuniones con la tutora, presentación del proyecto y elaboración del resto de partes de la documentación. Evidentemente todos los recursos definidos en realidad son el mismo recurso, el alumno con distintos roles. En la Figura 1 podemos ver la asignación de recursos por tarea, en la columna "Nombre de los recursos" y en la (Figura 3) se pueden apreciar los recursos con sus horas por tareas.

Id	Nombre del recurso	Trabajo
	Sin asignar	0 horas
1	Programador	280 horas
	Creación de interfaces	*6 horas*
	Programación	*233 horas*
	Pruebas	*40 horas*
	Entrega final	*1 hora*
2	Analista 1	48 horas
	Definir y documentar los requisitos de datos	*4 horas*
	Definir los requisitos tecnológicos y de plataforma	*4 horas*
	Documentar clases	*4 horas*
	Diagrama de actividades	*7 horas*
	Validar el análisis	*2 horas*
	Diseñar las interfaces gráficas	*19 horas*
	Refinar el diagrama de clases de diseño	*4 horas*
	Validar el diseño	*4 horas*
3	Jefe proyecto	111 horas
	Definición proyecto	*3 horas*
	Definir métodos de trabajo y documentación	*4 horas*
	Identificar el alcance y objetivos	*4 horas*
	Definir tareas y fechas	*2 horas*
	Crear el diagrama de Gantt	*4 horas*
	Documentar la propuesta del proyecto	*6 horas*
	Entregar la Propuesta Técnica	*1 hora*
	Revisión del tutor de la propuesta técnica	*1 hora*
	Entrega de propuesta técnica revisada	*1 hora*
	Redacción de informes quincenales	*6 horas*
	Redacción de la memoria técnica	*71 horas*
	Entrega de la memoria técnica	*1 hora*
	Preparación de la presentación oral	*6 horas*
	Presentación oral	*1 hora*
4	Analista 2	20 horas
	Revisar contexto y buscar información	*5 horas*
	Crear diagrama de Casos de Uso	*5 horas*
	Crear diagrama de Clases	*5 horas*
	Identificar y clasificar los usuarios	*5 horas*

Figura 3: Recursos con sus horas

Para asignar un coste a la hora a los recursos se ha hecho un pequeño estudio. Básicamente el estudio ha sido entrar en algunas webs de empleo, como son InfoJobs y Tecnoempleo y buscar ofertas de empleo con las características de nuestros recursos y proyecto. Las principales conclusiones que se han extraído son que a los recién graduados en Ingeniería, definidos en las ofertas como perfil Junior, sus sueldos varían entre 12.000 y 15.000 Euros brutos anuales. También hay que tener en cuenta que según las tecnologías de las ofertas también varían los salarios, siendo remarcable que aquellas relacionadas con el lenguaje PHP son más bajas que otras relacionadas con otros lenguajes como Java, .Net, Cobol. Otro factor es la distribución geográfica, siendo en la Comunidad Valenciana los sueldos inferiores a otras comunidades como Baleares, Madrid, Cataluña que a su vez publican más ofertas. Partiendo de este estudio se han decidido los siguientes salarios:

- Programador 6,50 Euros la hora que son 12.480 Euros anuales.
- Analista 7,50 Euros la hora que son 14.400 Euros anuales.
- Jefe del proyecto 8,50 Euros la hora que son 16.320 Euros anuales.

Como hay que tener en cuenta los impuestos por tener contratado el personal, se debe aplicar un aumento del 20 % al coste total de los recursos.

Recurso	Horas	€/hora	Total
Analista 1	20	7,5	150€
Analista 2	50	7,5	375€
Programador	280	6,5	1.820€
Jefe proyecto	100	8,5	850€
Total			3.195€
Impuesto	20 % del total		639€
Total final			3.834€

Cuadro 1: Coste en recursos

Al coste en recursos hay que añadir los costes de tipo material y tecnológicos:

- En la empresa:
  - Al alumno se le ha aportado un ordenador personal con un sistema operativo Windows Vista para que realice sus actividades durante los 3 meses de la estancia. Realizando un estudio de precios de renting de ordenadores, se ha estimado un coste de 35€al mes, es decir 105€.
  - La empresa ha hecho uso de un servidor público propio para que el alumno pudiera tener un servidor real donde realizar pruebas y la implantación del proyecto. Hay que tener en cuenta que el servidor lleva más proyectos internos de la empresa, como su propia página web. Estimación del coste de un mes por el uso de 1GB de RAM y 20GB memoria es de 20€.
  - Por contra el resto de tecnologías y programas utilizados son de uso gratuito. Com Atom como editor de textos, FileZilla como cliente FTP, XAMPP como distribución de Apache.

- Otros gastos a incluir son los derivados de la oficina como la luz, el agua y el material de oficina. Se estima un gasto aproximado de 2€la hora, que habiendo realizado 300 horas suman un total de 600€.

- Realización de la memoria:

  - Microsoft Project 2007, se ha usado en los ordenadores de la universidad, por tanto no supone ningún coste. Latex y Tex Live para la realización de la memoria, que no conlleva coste.
  - Para la realización de los distintos diagramas, se ha hecho uso de la herramienta de diagramas que proporciona Google Drive. También sin coste alguno.
  - Impresión y encuadernado de la memoria. 20€.

Gastos	Coste
PC sobremesa	105€
Servidor	60€
Oficina	600€
Impresión	20€
Total	785€

Cuadro 2: Otros gastos

Por tanto el coste final del proyecto, incluyendo todos los elementos es de 4.619€.

Recursos	3.834€
Otros gastos	785€
Total	4.619€

Cuadro 3: Tabla resumen de costes

3.4. Seguimiento del proyecto

Al ser un modelo en cascada no se puede iniciar una fase hasta que la anterior no esté terminada. Para llevar un control del proyecto por cada fase definida, se ha añadido una tarea de revisión y modificaciones al fin de la misma. El objetivo es tener bien definida y desarrollada cada fase.

En el transcurso del proyecto hubo una desviación causada por la falta definición de requisitos de datos. Para subsanar dicho problema se planteó unificar las tareas de diseño y la implementación usando un modelo de prototipos. El objetivo era que el cliente viera la funcionalidad de las interfaces con los datos de entrada y salida requeridos. Por otro lado, el modelo de prototipos genera una incertidumbre causada por la falta de previsión del número de iteraciones del prototipo hasta satisfacer al cliente. Se planteó por tanto, dar prioridad al Backend, dejando la implementación y el diseño del Frontend como las tareas finales del proyecto. Efectivamente el Frontend se quedó sin realizar solo pudiendo hacer el diseño y las interfaces pero sin funcionalidad y conexión a la base de datos.

Capítulo 4

Análisis del proyecto

En esta etapa se realiza un estudio de los requisitos del sistema con el objetivo de entender qué debe hacer el sistema. Inicialmente la empresa debería aportar los requisitos funcionales y de datos para realizar un análisis del mismo, para que en etapas posteriores poder crear la solución informática más adecuada.

Por otro lado como herramienta para el análisis se ha hecho uso del Lenguaje de Modelado Unificado ([8]) de OMG. Su finalidad es ayudar a especificar, visualizar y documentar modelos de sistemas software, incluyendo su estructura y su diseño, de manera que cumpla con todos sus requisitos.

Realizando esta documentación se constató que los requisitos de datos no estaban del todo claros o bien definidos. Para hacer una correcta documentación se realizó parte del Capítulo 5 con la finalidad de adquirir los requisitos de datos adecuados. Una vez identificados y la empresa satisfecha con ellos, se volvió a este capítulo para finalizar la documentación de los mismos.

En la Sección 4.1 se representan los casos de uso y se documentan los requisitos del sistema a partir de ellos. Posteriormente en la Sección 4.2 se detallan los procesos de negocio y sus actividades mediante diagramas de actividades.

4.1. Requisitos del proyecto

Como se comenta en la introducción al capítulo se ha hecho uso del lenguaje ULM para ayudar a analizar el sistema y documentar tanto los requistos funcionales como los de datos.

4.1.1. Casos de uso

Los casos de uso son una ayuda para comprender de forma sencilla el comportamiento del sistema mediante su representación gráfica. A continuación se exponen los distintos casos, primero los casos que representa el sistema completo y posteriormente se han sustraído los casos más complejos en casos más sencillos para mejorar el análisis.

Casos de uso general del proyecto

El la Figura 4, muestra el sistema completo o general del proyecto, donde hay ciertos casos que se han sintetizado o resumido dada su complejidad, los llamados "abstract".

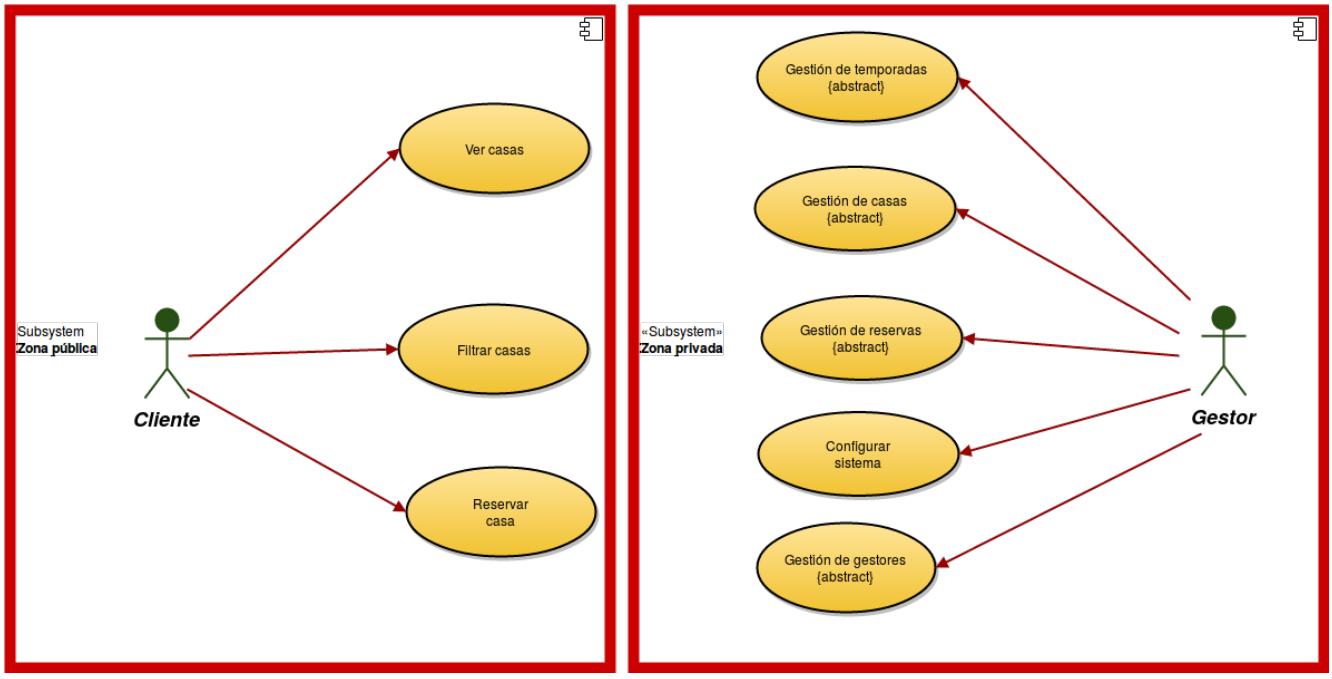

Figura 4: Casos de uso general del proyecto

Casos de uso de gestión de gestores

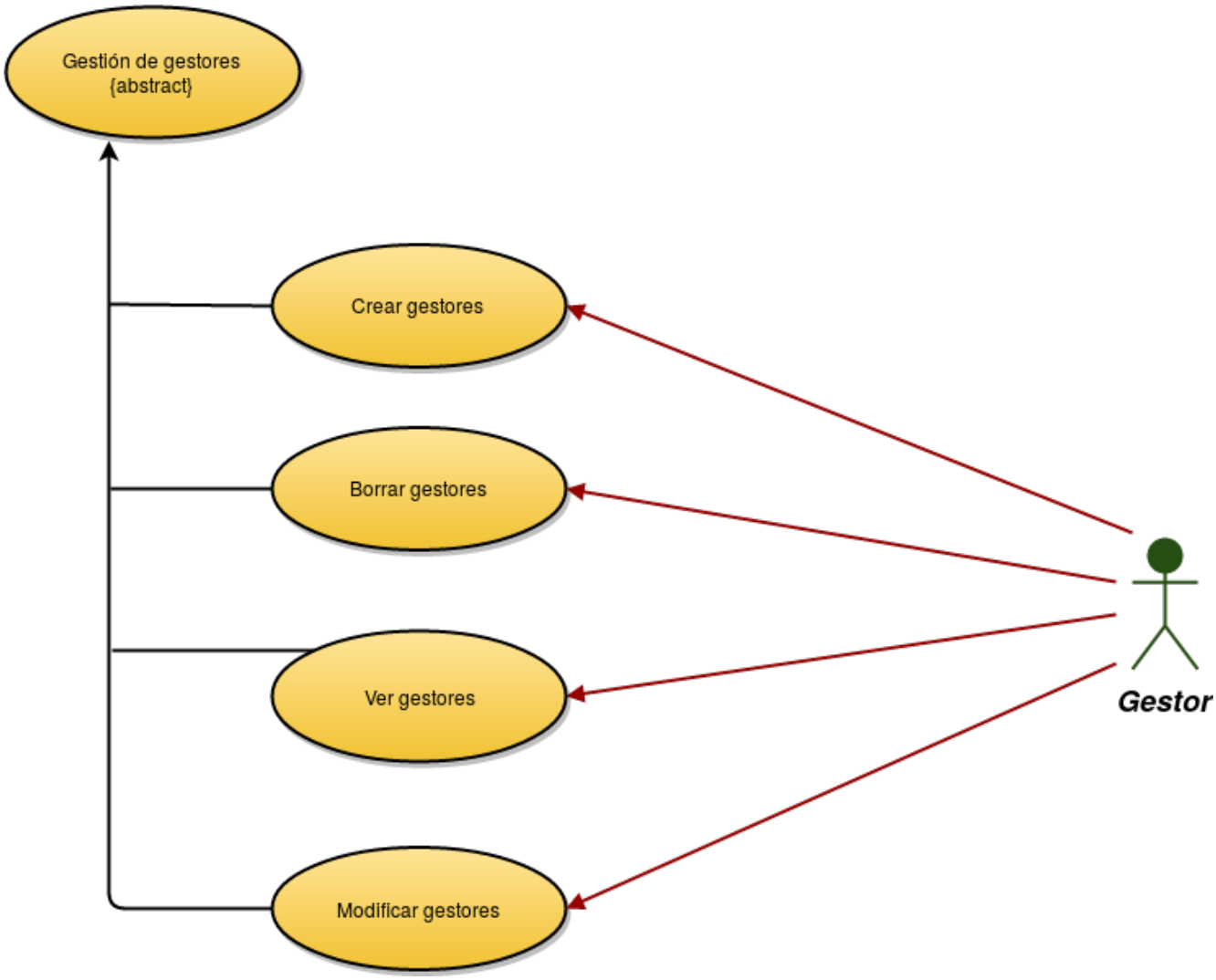

Figura 5: Casos de uso de gestión de gestores

Casos de uso de gestión de temporadas

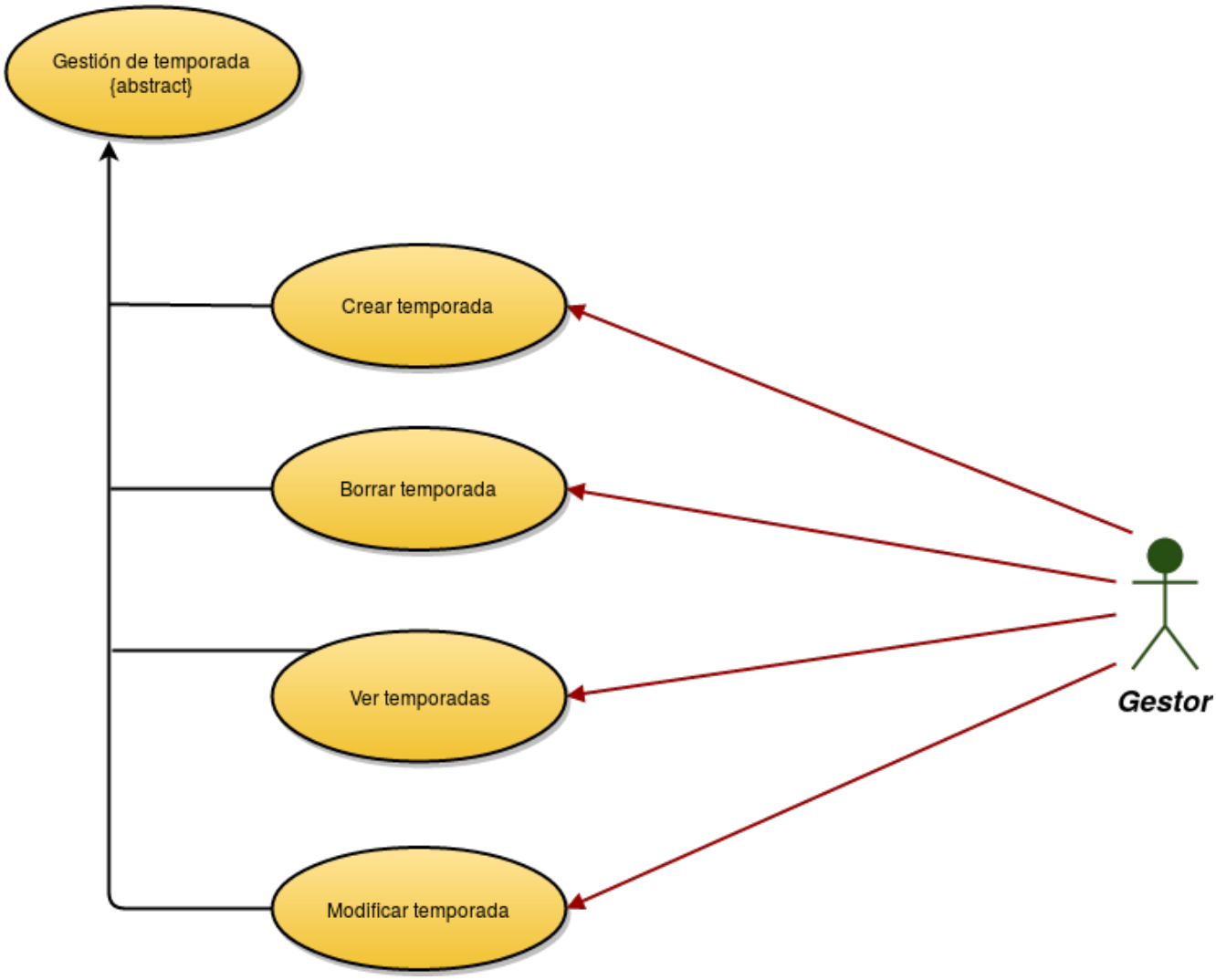

Figura 6: Casos de uso de gestión de temporadas

Casos de uso de gestión de casas

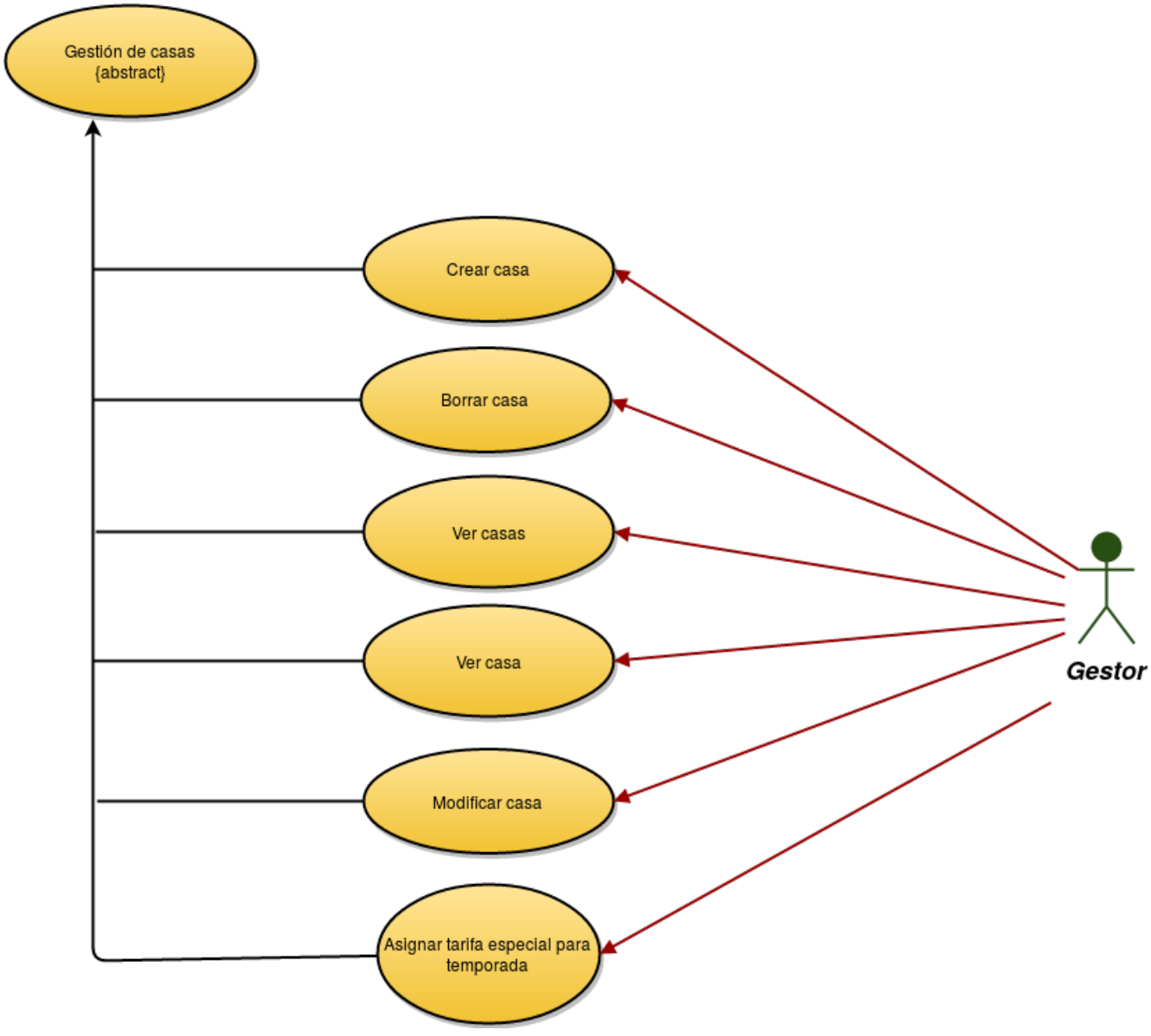

Figura 7: Casos de uso de gestión de casas

Casos de uso de gestión de reservas

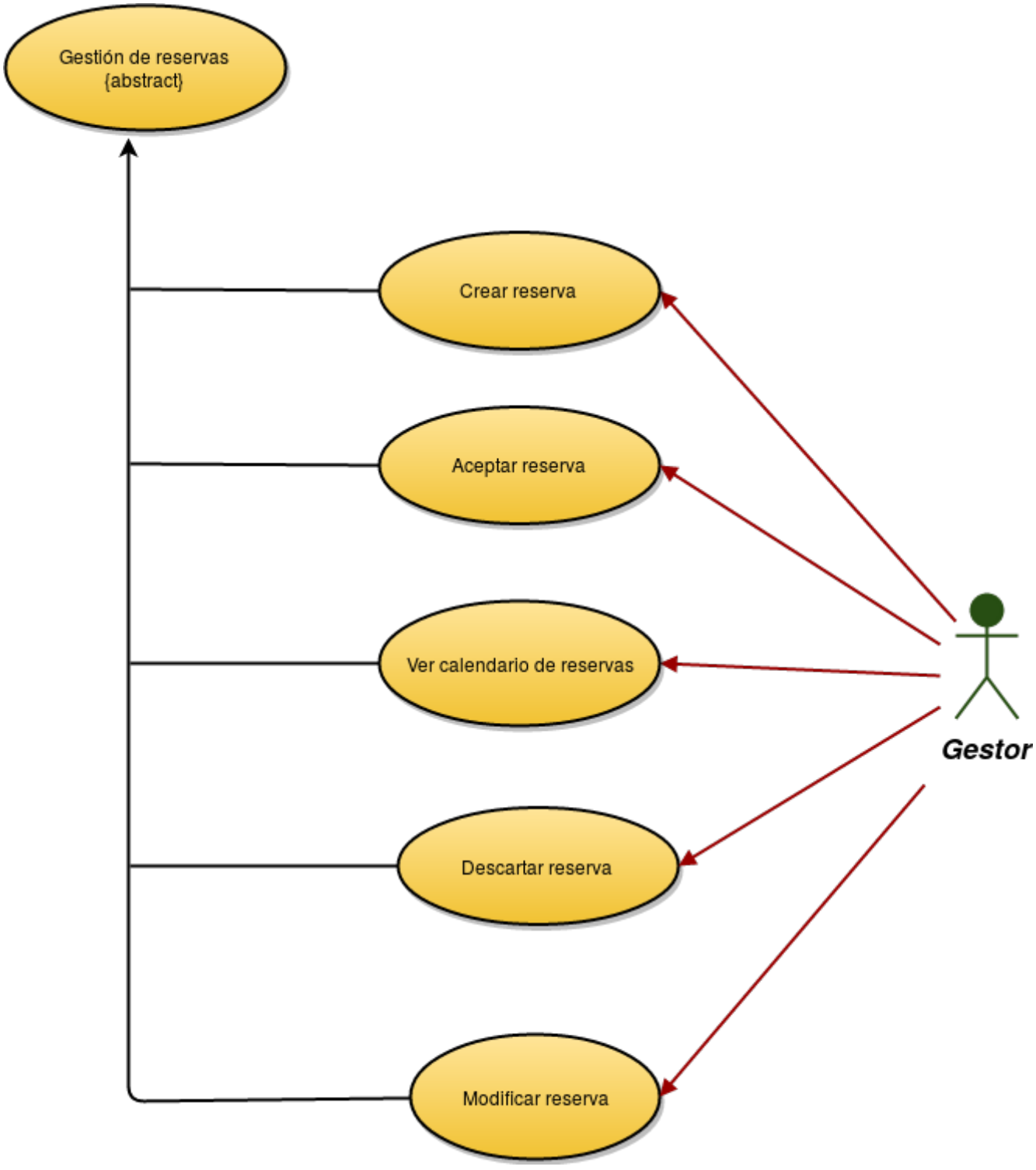

Figura 8: Casos de uso de gestión de reservas

4.1.2. Requistos funcionales

Nombre	Configurar sistema (Figura 4)
Fuentes	Interfaz de configuración y responsable de prácticas.
Descripción	Parametrización de los datos de configuración del sistema.
Nivel	Tarea secundaria.
Actor	Gestor.
Requisitos de datos	Configuración del sistema (Tabla 27)
Precondición	El sistema ya debe tener unos valores por defecto.
Trigger	Modificación de los parámetros de configuración.
Secuencia normal	Acción
1. 2. 3. 4. 5.	Selección de la opción de “configuración”. Acceso al formulario de configuración. Modificación de los datos. Selección del botón modificar. Modificación de los datos.
Excepciones	
1.	Los campos numéricos sólo admiten valores numéricos.

Cuadro 4: Configurar sistema

Nombre	Ver casas Figura 4
Fuentes	Responsable de prácticas.
Descripción	Ver todas las casas y sus disponibilidades.
Nivel	Tarea principal.
Actor	Cliente.
Requisitos de datos	Casa (Tabla 34), Reserva (Tabla 35), Periodo de cierre (Tabla 33)
Precondición	- Filtar aquellas casas no publicadas.
Trigger	Recuperar todos los datos de la casa, sus periodos tanto de cierre como de periodo de cierre.
Secuencia normal	Acción
1. Excepciones	Acceso al listado de casas. -

Cuadro 5: Ver casas. (Cliente)

Nombre	Filtrar casas (Figura 4)
Fuentes	Responsable de prácticas.
Descripción	Listado de las casas donde se puede realizar un filtrado por una serie de parámetros.
Nivel	Tarea principal.
Actor	Cliente.
Requisitos de datos	Casa (Tabla 34), Reserva (Tabla 35), Periodo de cierre (Tabla 33), Tipo casa (Tabla 29), Característica (Tabla 31), Actividad (Tabla 30)
Precondición	-
Trigger	Recuperar todas las casas que cumplan con los parámetros introducidos por el cliente de aquellas casas en estado publicadas.
Secuencia normal	Acción
1. 2. 3. 4.	Acceso al interfaz. Rellenar campos del formulario. Seleccionar el botón de filtrar. Listar todas las casas y sus disponibilidades de aquellas que cumplan con las condiciones del formulario.
Excepciones :	
1.	Solo se muestran casas en estado publicadas.

Cuadro 6: Filtrar búsqueda.

Nombre	Reservar casa (Figura 4)
Fuentes	Responsable de prácticas.
Descripción	Crear una nueva reserva.
Nivel	Tarea principal.
Actor	Cliente.
Requisitos de datos	Casa (Tabla 34), Reserva (Tabla 35)
Precondición	Tener una casa seleccionada, sus días libres y sus tarifas.
Trigger	Insertar los datos de una reserva relacionada con una casa.
Secuencia normal	Acción
1.	Selección de una casa desde el interfaz de filtrado de casas.
2.	Seleccionar el botón reservar.
3.	Acceso al formulario de alta de una reserva.
4.	Rellenar los datos del formulario.
5.	Seleccionar el botón reservar.
5.	Mostrar las condiciones de la reserva.
Excepciones	
1.	Solo se muestran las casas en estado publicadas.
2.	La fecha de entrada no puede ser mayor o igual a la fecha de salida.
3.	Rellenar los campos obligatorios.
4.	Formato de los campos deben ser los correctos.

Cuadro 7: Reservar casa (cliente).

Nombre	Crear gestor (Figura 5)
Fuentes	Interfaz de alta de gestor y responsable de prácticas.
Descripción	Crear un nuevo administrador para el sistema.
Nivel	Tarea secundaria.
Actor	Gestor.
Requisitos de datos	Gestor (Tabla 28)
Precondición	Debe existir al menos un usuario gestor.
Trigger	Inserta nuevo gestor en el sistema.
Secuencia normal	Acción
1. 2. 3. 5.	Acceso al listado de gestores Acceso al formulario de alta de gestores. Introducción de los datos de gestores. Selección del botón crear.
Excepciones	
1. 2. 3.	Comprobación de los campos formulario no vacíos. Dos campos password para verificar que se ha introducido correctamente el password. Que no existan espacios en blanco en el nombre del gestor.

Cuadro 8: Crear gestor

Nombre	Borrar gestor (Figura 5)
Fuentes	Interfaz de listado de gestores y responsable de prácticas.
Descripción	Elimina del sistema un gestor.
Nivel	Tarea secundaria.
Actor	Gestor.
Requisitos de datos	Gestor (Tabla 28)
Precondición	El gestor no puede borrarse a sí mismo.
Trigger	Elimina del sistema los datos del gestor.
Secuencia normal	Acción
1. 2. Excepciones	Acceso al listado de los gestores. Seleccionar el botón borrar del gestor deseado.
1. 2.	El sistema no se debe quedar sin gestor. Un gestor no puede borrar su usuario.

Cuadro 9: Borrar gestor

Nombre	Ver gestores (Figura 5)
Fuentes	Interfaz de listado de gestores y responsable de prácticas.
Descripción	Listado de todos los gestores del sistema.
Nivel	Tarea secundaria.
Actor	Gestor.
Requisitos de datos	Gestor (Tabla 28)
Precondición	-
Trigger	Recuperar todos los usuarios del sistema.
Secuencia normal	Acción
1.	Acceso al listado de gestores.
Excepciones	
	No se permite ver el password.

Cuadro 10: Ver gestores

Nombre	Modificar gestor (Figura 5)
Fuentes	Interfaz de modificar al gestor y responsable de prácticas.
Descripción	Modifica los datos de un usuario gestor existente.
Nivel	Tarea secundaria.
Actor	Gestor.
Requisitos de datos	Gestor (Tabla 28)
Precondición	Debe existir el usuario gestor en sistema.
Trigger	Modifica los datos de un usuario gestor en el sistema.
Secuencia normal	Acción
1. 2. 3. 5.	Acceso al listado de gestores. Acceso al formulario de modificación del gestor. Modificación de los datos del gestor. Selección del botón modificar.
Excepciones	
1. 2. 3. 4.	Comprobación de los campos del formulario no estén vacíos. Dos campos password, para verificar que se ha introducido correctamente el password. Que no existan espacios en blanco en el nombre del usuario. No se permite ver los datos de password original.

Cuadro 11: Modificar gestor

Nombre	Crear temporada (Figura 6)
Fuentes	Interfaz de alta de temporada y responsable de prácticas.
Descripción	Crear una nueva temporada para el sistema.
Nivel	Tarea secundaria.
Actor	Gestor.
Requisitos de datos	Temporada (Tabla 32)
Precondición	Recuperar el día actual.
Trigger	Inserta una nueva temporada en el sistema.
Secuencia normal	Acción
1. 2. 3. 5.	Acceso al listado de temporadas. Acceso al formulario de alta de temporadas. Introducción de los datos de una nueva temprada. Selección del botón crear.
	Excepciones :
1. 2. 3. 4.	Comprobación de los campos obligatorios. Las fechas de la temporada deben ser a futuro. La fecha de inicio de la temporada no puede ser a mayor que la de fin. El nombre debe ser único en sistema.

Cuadro 12: Crear temporada

Nombre	Borrar temporada (Figura 6)
Fuentes	Interfaz de listado de temporadas y responsable de prácticas.
Descripción	Elimina del sistema una temporada.
Nivel	Tarea secundaria.
Actor	Gestor.
Requisitos de datos	Temporada (Tabla 32)
Precondición	La temporada debe estar en el pasado.
Trigger	Elimina del sistema los datos de la temporada.
Secuencia normal	Acción
1. 2.	Acceso al listado de temporadas. Seleccionar el botón borrar.
Excepciones	
1.	Si la temporada es a futuro no debe estar relacionada con una casa del sistema.

Cuadro 13: Borrar temporada.

Nombre	Ver temporadas (Figura 6)
Fuentes	Interfaz de listado de tempradas y responsable de prácticas.
Descripción	Listado de todas las temporadas que hay en el sistema.
Nivel	Tarea secundaria.
Actor	Gestor.
Requisitos de datos	Temporada (Tabla 32)
Precondición	Diferenciar las temporadas pasadas y futuras.
Trigger	Recuperar todas las temporadas del sistema.
Secuencia normal	Acción
1.	Acceso al listado de temporadas.
Excepciones	-

Cuadro 14: Ver temporada.

Nombre	Modificar temporada (Figura 6)
Fuentes	Interfaz modificar temporada y responsable de prácticas.
Descripción	Modificar temporada ya existente en el sistema.
Nivel	Tarea secundaria.
Actor	Gestor.
Requisitos de datos	Temporada (Tabla 32)
Precondición	Recuperar el día actual.
Trigger	Modifica la temporada en el sistema.
Secuencia normal	Acción
1.	Acceso al listado de temporadas.
2.	Acceso al formulario de modificación de temporadas.
3.	Modificación de los datos de la temporada.
5.	Selección del botón modificar.
Excepciones	
1.	Comprobación de los campos obligatorios.
2.	Las fechas de la temporada deben ser a futuro.
3.	La fecha de inicio de la temporada no puede ser a mayor que la de fin.
3.	El nombre debe ser único en sistema.

Cuadro 15: Modificar Temporada.

Nombre	Crear casa (Figura 7)
Fuentes	Interfaz de alta de casa y responsable de prácticas.
Descripción	Crear una nueva casa para el sistema.
Nivel	Tarea principal.
Actor	Gestor.
Requisitos de datos	Casa (Tabla 34), Periodo de cierre (Tabla 33), Tipo de casa (Tabla 29), Actividad (Tabla 30) y Característica (Tabla 31)
Precondición	Recuperar el día actual.
Trigger	Inserta una nueva casa en el sistema y sus datos asociados como el tipo casa, actividad, características y periodo de cierre.
Secuencia normal	Acción
1.	Acceso al formulario de alta de casa.
3.	Introducción de los datos de la casa.
4.	Relacionarla con el tipo de casa.
5.	Relacionarla con sus características.
6.	Relacionarla con sus actividades.
7.	Crear periodos de cierre para la casa.
5.	Selección del botón crear.
Excepciones	
1.	Comprobación de los campos obligatorios.
2.	Las fechas del periodo deben ser a futuro.
3.	La fecha de inicio del periodo no puede ser a mayor que la de fin.
3.	El nombre debe ser único en sistema.
4.	Los campos del tipo modena, deben ser numéricos y con dos decimales.

Cuadro 16: Crear casa.

Nombre	Borrar casa (Figura 7)
Fuentes	Interfaz de listado de casas para borrar y responsable de prácticas.
Descripción	Elimina del sistema la casa.
Nivel	Tarea principal.
Actor	Administrador.
Requisitos de datos	Casa (Tabla 34)
Precondición	La casa no debe tener ninguna reserva a futuro.
Trigger	Elimina del sistema los datos de la casa.
Secuencia normal	Acción
1.	Acceso al listado de casas para borrar.
2.	Seleccionar el botón borrar.
Excepciones	
1.	La casa no debe tener ninguna reserva a futuro relacionada en el sistema.

Cuadro 17: Borrar casa.

Nombre	Ver casas (Figura 7) (Figura 6)
Fuentes	Interfaz de listado casas y responsable de prácticas.
Descripción	Listado de todas las casas que hay en el sistema.
Nivel	Tarea secundaria.
Actor	Gestor.
Requisitos de datos	Casa (Tabla 34)
Precondición	-
Trigger	Recuperar todas las casas del sistema.
Secuencia normal	Acción
1.	Acceso al listado de casas.
Excepciones	-

Cuadro 18: Ver casa.

Nombre	Ver casa (Figura 7)
Fuentes	Interfaz de ver casa y responsable de prácticas.
Descripción	Ver todos los datos relacionados con la casa sin la posibilidad de hacer ninguna acción, para mantener su integridad de datos.
Nivel	Tarea principal.
Actor	Gestor.
Requisitos de datos	Casa (Tabla 34)
Precondición	-
Trigger	Recuperar todos los datos de la casa, sus periodos de cierre, sus actividades, características y tipo de casa.
Secuencia normal	Acción
1. 2.	Acceso al listado de casas. Seleccionar el botón de ver casa.
Excepciones	-

Cuadro 19: Ver casa.

Nombre	Modificar casa.
Fuentes	Interfaz de modificar casa y responsable de prácticas.
Descripción	Modifica los datos de la casa y sus datos relacionados como el tipo de casa, sus características, actividades y sus periodos de cierre.
Nivel	Tarea principal.
Actor	Gestor.
Requisitos de datos	Casa (Tabla 34), Periodo de cierre (Tabla 33), Tipo de casa (Tabla 29), Actividad (Tabla 30) y Característica (Tabla 31)
Precondición	Recuperar el día actual.
Trigger	
1.	Recuperar todos los datos de la casa, sus periodos de cierre a futuro, sus actividades, características y tipo de casa.
2.	Modificar los datos de la casa, sus periodos de cierre, sus actividades, características y tipo de casa.
Secuencia normal	Acción
1.	Acceso al listado de casas.
2.	Acceso al formulario de modificar la casa.
3.	Modificar datos de la casa.
4.	Modificar relación con el tipo de casa.
5.	Modificar sus características.
6.	Modificar sus actividades.
7.	Modificar los periodos de cierre para la casa.
8.	Selección del botón modificar.
Excepciones	
1.	Comprobación de los campos obligatorios.
2.	Las fechas del periodo deben ser a futuro.
3.	La fecha de inicio del periodo no puede ser a mayor que la de fin.
3.	En nombre debe ser único en sistema.
4.	Los campos del tipo modena, deben ser numéricos y con dos decimales.

Cuadro 20: Modificar casa.

Nombre	Asignar una tarifa especial a temporada (Figura 7)
Fuentes	Interfaz tarifas por temporada y responsable de prácticas.
Descripción	Asigna un tarifa especial por casa y temporada.
Nivel	Tarea principal.
Actor	Gestor.
Requisitos de datos	Casa (Tabla 34) y Temporada (Tabla 32)
Precondición	Temporadas solo futuras y si no hay tarifa especial asignar la tarifa base de la casa.
Trigger	
1.	Recuperar todas las casas, todos los periodos y sus posibles tarifas.
2.	Modificar o crear una tarifa y relacionarla con una casa y una temporada específica.
Secuencia normal	Acción
1.	Acceso al interfaz de tarifas por temporada.
2.	Asignar importe a la temporada por la casa.
3.	Seleccionar el botón de modificar.
Excepciones	
1	Los campos del tipo modena, deben ser numéricos y con dos decimales.

Cuadro 21: Asignar tarifa especial a temporada.

Nombre	Ver calendario de reservas (Figura 8)
Fuentes	Interfaz de listado de reservas y responsable de prácticas.
Descripción	Un calendario donde se muestra las reservas y los periodos de cierre de forma gráfica.
Nivel	Tarea principal.
Actor	Gestor.
Requisitos de datos	Casa (Tabla 34), Periodo de cierre (Tabla 33) y Reserva (Tabla 35)
Precondición	Recuperar la fecha actual.
Trigger	Recuperar todas las casa activas y sus reservas y los periodos de cierre con sus fechas a futuro.
Secuencia normal	Acción
1.	Acceso al interfaz.
2.	Listar todas las casas con sus reservas y periodos de cierre.
Excepciones	-

Cuadro 22: Ver calendario de reservas.

Nombre	Aceptar reserva (Figura 8)
Fuentes	Formulario de modificar reserva y el responsable de prácticas.
Descripción	Se acepta una reserva, es decir se pasa de estado pendiente a aceptada.
Nivel	Tarea principal.
Actor	Gestor.
Requisitos de datos	Casa (Tabla 34) y Reserva (Tabla 35)
Precondición	Tener una reserva seleccionada en estado pendiente.
Trigger	Modificar el estado de la reserva a aceptada.
Secuencia normal	Acción
1. 2. 3.	Acceso al interfaz de listado de reservas y seleccionar una en estado pendiente Acceder al formulario de modificar reserva. Seleccionar el botón de aceptar reserva.
Excepciones	-

Cuadro 23: Aceptar reserva reserva.

Nombre	Descartar reserva (Figura 8)
Fuentes	Formulario de modificar reserva y el responsable de prácticas.
Descripción	Se elimina una reserva del sistema.
Nivel	Tarea principal.
Actor	Gestor.
Requisitos de datos	Casa (Tabla 34) y Reserva (Tabla 35)
Precondición	Tener una reserva seleccionada.
Trigger	Borrar reserva.
Secuencia normal	Acción
1. 2. 3.	Acceso al interfaz de listado de reservas y seleccionar una en estado pendiente. Acceder al formulario de modificar una reserva. Seleccionar el botón de denegar reserva.
Excepciones	-

Cuadro 24: Descartar reserva.

Nombre	Modificar reserva (Figura 8)
Fuentes	Formulario de modificar reserva y el responsable de prácticas.
Descripción	Se modifican los datos de una reserva.
Nivel	Tarea principal.
Actor	Gestor.
Requisitos de datos	Casa (Tabla 34), Temporada (Tabla 32) y Reserva (Tabla 35)
Precondición	Tener una reserva seleccionada.
Trigger	Modificar los datos de la reserva.
Secuencia normal	Acción
1.	Acceso al interfaz de listado de reservas y seleccionar una.
2.	Acceder al formulario de modificar reserva.
3.	Modificar los datos de la reserva.
3.	Seleccionar el botón de modificar reserva.
Excepciones :	
1.	La fecha inicio si está en pasado no se puede modificar.
2.	La fecha inicio no puede ser mayor a la fecha fin.
3.	La fecha inicio no puede ser menos a la fecha actual.
4.	Campos obligatorios.
5.	Campos del tipo importe deben tener dos decimales.

Cuadro 25: Modificar reserva.

Nombre	Crear reserva (Figura 8)
Fuentes	Interfaz de alta una reserva y el responsable de prácticas.
Descripción	Crear una nueva reserva.
Nivel	Tarea principal.
Actor	Gestor.
Requisitos de datos	Casa (Tabla 34), Temporada (Tabla 32) y Reserva (Tabla 35)
Precondición	Recuperar fecha actual.
Trigger	Insertar los datos de una reserva relacionada con una casa.
Secuencia normal	Acción
1.	Desde el interfaz del calendario de resevas.
2.	Seleccionar el botón nueva reservar.
3.	Acceso al formulario de alta de una reserva.
4.	Rellenar los datos del formulario.
5.	Mostrar los datos de las tarifas según sus fechas.
6.	Seleccionar el botón reservar.
Excepciones :	
1.	Solo se muestran casas en estado activas.
2.	La fecha de entrada no puede ser mayor o igual a la fecha de salida.
3.	Rellenar los campos obligatorios.
4.	El formato de los campos deben ser los correctos.
5.	No solapar fechas con otras reservas.

Cuadro 26: Crear reserva (Gestor).

4.1.3. Requistos de datos

Nombre	Configuración del sistema.
Requisito asociado	Configurar sistema (Tabla 4)
Fuente	Interfaz de configuración y responsable de prácticas.
Datos específicos	fin de semana, máximo días reservables, fecha límite reserva.
Importancia	Alta.
Comentarios	Fin de semana no admite nulos.

Cuadro 27: Configuración del sistema

Nombre	Gestor
Requisito asociados	Crud gestión de gestores ver (Tabla 10), crear (Tabla 8), modificar (Tabla 11) y borrar (Tabla 9)
Fuente	Interfaces de login y las relacionadas con el crud de gestor más el responsable de prácticas.
Datos específicos	Nombre usuario, clave de usuario.
Importancia	Alta.
Comentarios	
	Todos los datos son obligatorios. No puede haber dos nombres de usuario iguales en el sistema.

Cuadro 28: Gestor

Nombre	Tipo de casa.
Requisitos asociados	Filtar casa (Tabla 6), Crear casa (Tabla 16) y modificar casa (Tabla 20).
Fuente	Interfaces de alta tipo de casa, de listados de tipos de casas y de modificar de tipos de casas más responsable de prácticas.
Datos específicos	Nombre, nombre descripción.
Importancia	Alta.
Comentarios	No puede haber dos nombres iguales en el sistema.

Cuadro 29: Tipo de casa

Nombre	Actividad.
Requisitos asociados	Filtar casa (Tabla 6), Crear casa (Tabla 16) y modificar casa (Tabla 20).
Fuente	Interfaces del crud de gestíon de actividades.
Datos específicos	Nombre, nombre descripción.
Importancia	baja.
Comentarios	No puede haber dos nombres iguales en el sistema.

Cuadro 30: Actividad

Nombre	Carasterística
Requisitos asociados	Filtar casa (Tabla 6), Crear casa (Tabla 16) y modificar casa (Tabla 20).
Fuente	Interfaces del crud de gestíon de características.
Datos específicos	Nombre, nombre descripción.
Importancia	baja.
Comentarios	No puede haber dos nombres iguales en el sistema.

Cuadro 31: Característica

Nombre	Temporada.
Requisitos asociados	Los relacionados con el crud de gestión de temporadas como són crear (Tabla 12), borrar (Tabla 13) , ver (Tabla 14), modificar (Tabla 15) temporada más el caso de asignar tarifa especial a temporada Tabla 21.
Fuente	Interfaces de alta temporada, listado de temporadas y modificar temporada más el responsable de prácticas.
Datos específicos	Nombre, descripción, fecha de inicio, fecha de fin, días mínimos para una reserva, tarifas mínimo en días de una reserva.
Importancia	Alta.
Comentarios	
1.	No puede haber dos nombres iguales en el sistema.
2.	La fecha de inicio no puede mayor que la fecha de fin.
3.	"Días mínimos" no puede ser mayor a los días del periodo.
4.	"Tarifas mínimo en días" no puede ser mayor a los días del periodo.

Cuadro 32: Temporada

Nombre	Periodo de cierre.
Requisitos asociados	Filtar casa (Tabla 6), Ver la calendario de reservas (Tabla 22), Crear casa (Tabla 16) y Modificar casa (Tabla 20).
Fuente	Interfaces de alta casa, ver casa y modificar casa más el responsable de prácticas.
Datos específicos	Nombre, nombre descripción, fecha de inicio y fecha de fin.
Importancia	Alta.
Comentarios	
1.	La fecha de inicio no puede mayor que la fecha de fin.

Cuadro 33: Periodo de cierre.

Nombre	Casa
Requisitos asociados	Asignar tarifa especial a temporada (Tabla 21), ver casas (Tabla 18), ver casa (Tabla 19), modificar casa (Tabla 20), crear casa (Tabla 16).
Fuente	Interfaces de tarifas por periodo, alta casa, listado de casas, ver casa y modificar casa más el Responsable de prácticas.
Datos específicos	Nombre, tipo casa, activa, publicada, tarifa base, tarifa fin de semana, descuento semana, descuento mes, número de camas, número de personas, numero de habitaciones, número de baños, lista de actividades, listas de características, listas de periodos de cierre, lista de tarifas por temporada.
Importancia	Alta.
Comentarios	
1. 2.	No puede haber dos nombres iguales en el sistema. Si no está activa no puede ser publicada.

Cuadro 34: Casa.

Nombre	Reserva
Requisitos asociados	Los relacionados con el crud de reservas para el gestor como son Crear reserva (Tabla 26), Modificar reserva (Tabla 25), aceptar reserva (Tabla 23), descartar reserva (Tabla 24) y ver calendario de reservas (Tabla 22). Hay que incluir los requisitos relacionados con el cliente como son ver casa (Tabla 5), filtrar casas (Tabla 6) y reservar casa (Tabla 7).
Fuente	Interfaces alta reserva y de modificar reserva más el responsable de prácticas.
Datos específicos	Fecha inicio, fecha fin, casa relacionada, tarifa calculada, tarifa acordada, nombre cliente, email cliente, número de teléfono y comentario cliente.
Importancia	Alta.
Comentarios	No se solapen reservas para una misma casa.

Cuadro 35: Reserva.

4.2. Diagrama de actividades

En este punto se ha seguido analizando y modelando el sistema. Otra vez se ha decidido usar una herramienta dentro del lenguaje UML, como son los diagramas de actividades, donde se ha analizado el sistema desde una perspectiva de procesos distinguiendo las acciones de una forma secuencial. Se ha definido un diagrama general del sistema Figura 10, destacando los actores del sistema y sus actividades. Hay que remarcar que la mayoría de las actividades del administrador son más complejas, como es el caso de gestionar usuarios Figura 9 , pero se ha decidido no realizar el modelado de dichas actividades con el fin de tener una visión global del sistema sin entrar al detalle.

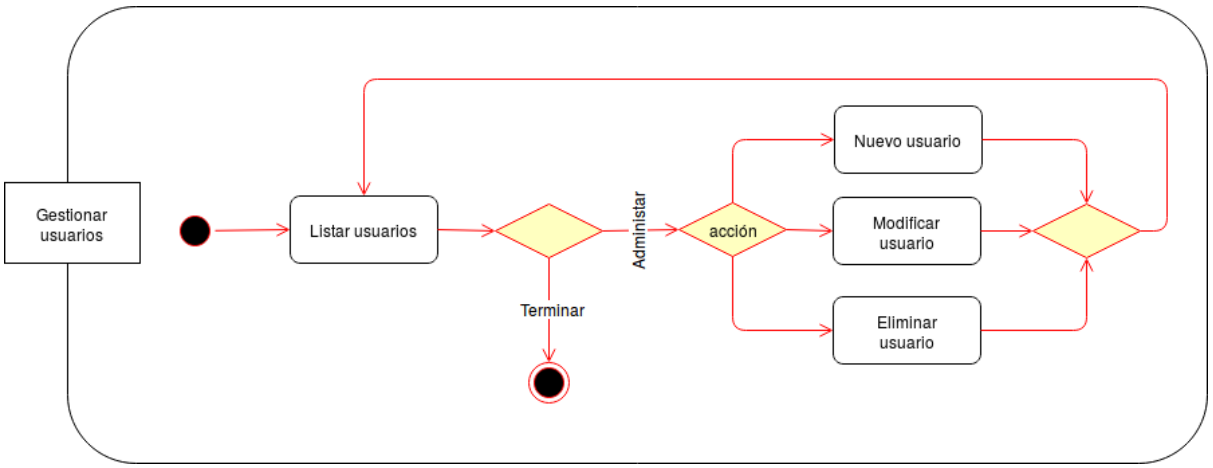

Figura 9: Diagrama de actividades 'gestionar usuarios'

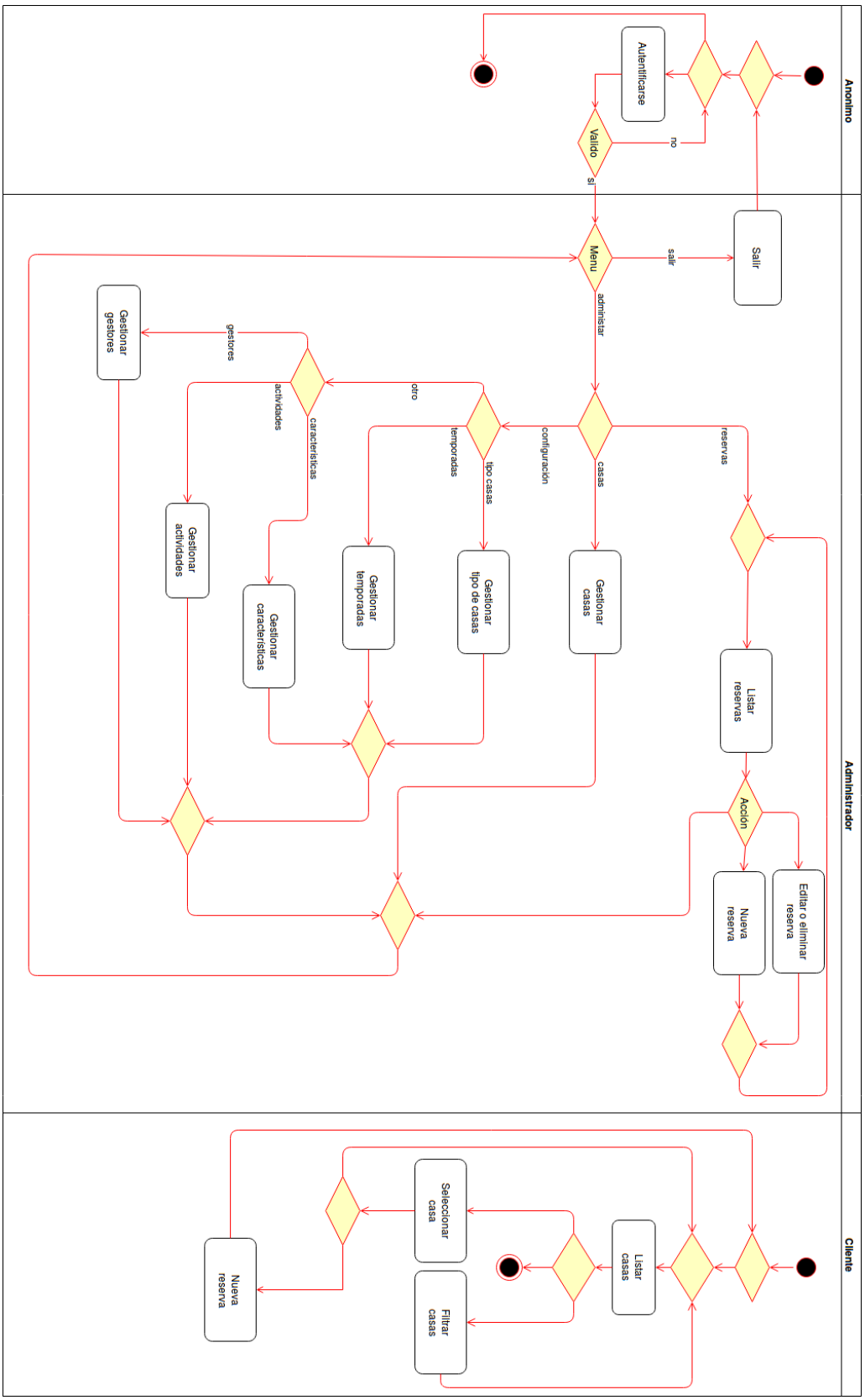

Figura 10: Diagrama de actividades del sistema

Capítulo 5

Diseño e implementación del proyecto

Esta sección integra la descripción del desarrollo del diseño e implementación. Esta integración de las dos fases fue consecuencia de un contratiempo en el análisis; el cliente tenía claramente identificada la funcionalidad del producto, las tecnologías necesarias para su desarrollo y un modelo interfaz seleccionado, pero por otro lado no tenía los requisitos de datos bien identificados.

Es decir, nos encontrábamos en la siguiente situación:

- Como se ha comentado en capítulos anteriores, el interfaz seleccionado por la empresa es AdminLITE, desarrollado en HTML, CSS, JavaScript, Bootstrap, el cual está repleto de ejemplos que se suelen usar en una gestión interna, por lo que este producto ofrece un base y una estructura para el diseño de nuestras interfaces.
- También está analizada la funcionalidad del sistema y su estructura de navegación a nivel de interfaces.
- Y por último, faltaba identificar los requisitos de datos para documentarlos y posteriormente poder realizar el diseño e implementación de la base de datos.

Partiendo de estos puntos se decidió integrar la adquisición de los requisitos de datos mediante el diseño de interfaces y su implementación en un mismo punto... ¿Cómo? Pues usando el modelo de Ingeniería del Software de prototipos.

En las siguientes secciones se describen las distintas partes que componen este capítulo de diseño e implementación del proyecto. Aunque los puntos no siguen un orden cronológico, se ha decido realizar la siguiente estructura ya que se considera más adecuada para su documentación. A continuación se detalla como son los roles de usuario Sección 5.1, la base de datos Sección 5.2, el modelo cliente-servidor con las tecnologías aplicadas Sección 5.3 y el modelo aplicado para su desarrollo Sección 5.5 así como el mapa de las interfaces Sección 5.6 y el apartado de pruebas Sección 5.7.

5.1. Usuarios

Como se ha comentado en el capítulo de análisis, en el proyecto existen dos roles a nivel de aplicación: el Gestor y el Cliente. Ahora desde el punto de vista del diseño y la implementación entran en escena otra serie de usuarios. Para aclarar posibles confusiones se ha incluído una descripción de cada uno de estos usuarios:

- Administrador del sistema. Es el encargado de administrar el servidor, su principal tarea es subir y bajar archivos y crear la estructura de ficheros necesaria. Hay que recordar que el alumno en prácticas no administraba ni la configuración de PHP ni la de Apache.
- Administrador de la base de datos. Como su nombre indica es administrador de la base de datos y se encarga de crear bases de datos y sus tablas así como nuevos usuarios. Como se verá en la sección Sección 5.2 por motivos de seguridad se han creado dos nuevos usuarios con acceso a la base de datos de la aplicación:
  - Usuario cliente, que está relacionado con el rol del cliente.
  - Usuario gestor, que está relacionado con el rol del gestor.

5.2. Base de datos

Para la realización del diseño de la base de datos se ha elegido el modelo relacional, donde la estructura de datos básica es la tabla y la relación. En los siguientes puntos se describirán el resultado de aplicar las distintas etapas del diseño de una base de datos relacional al sistema.

5.2.1. Diseño conceptual

En esta etapa se decribe la información del sistema mediante el esquema conceptual. A medida que se construye se descubre la semántica (significado) de los datos del sistema y se especifican las entidades, atributos y relaciones.

En un primer momento se muestra el diseño gráfico de la base de datos mediante el diagrama entidad relación (Figura 11).

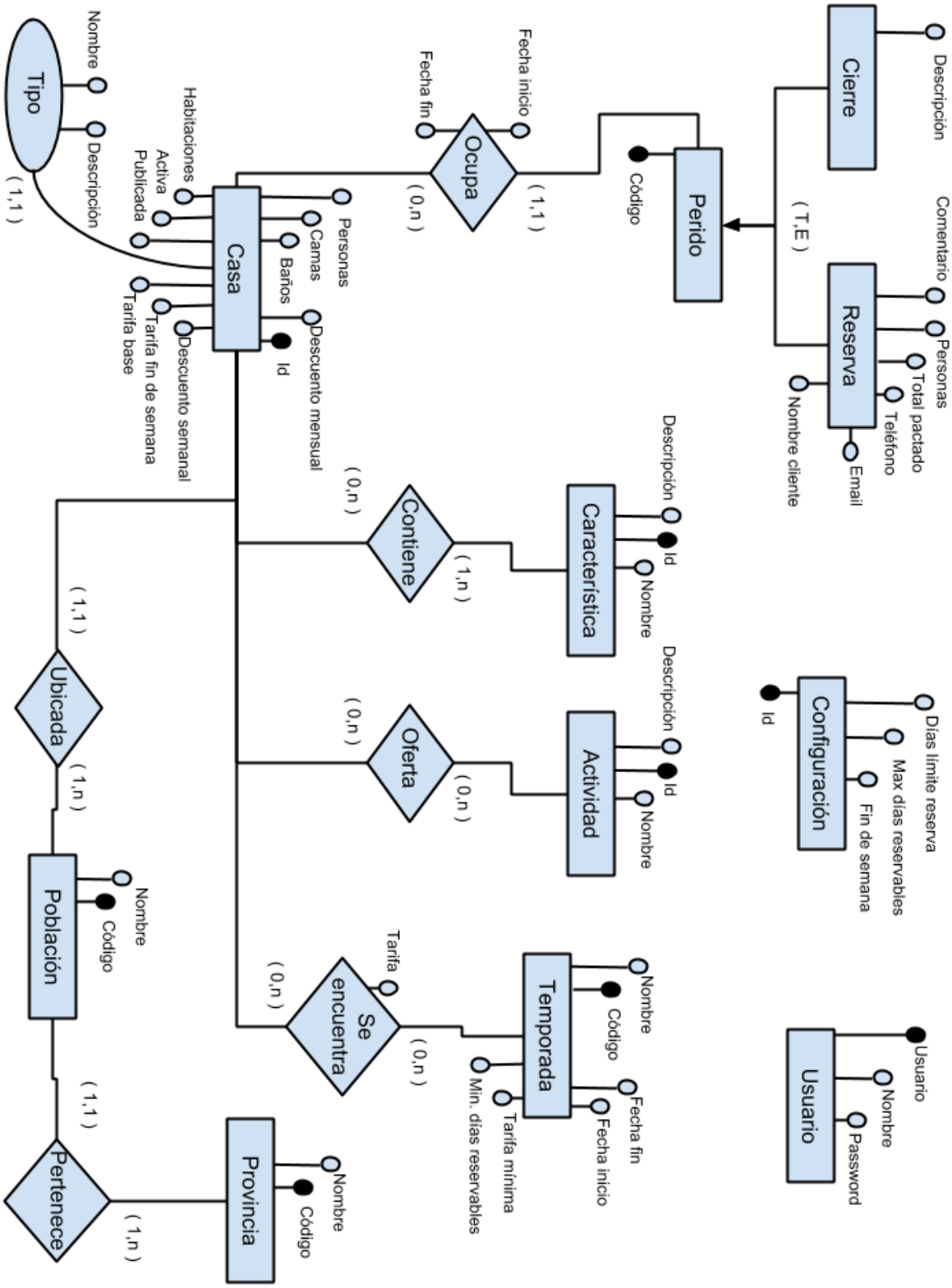

Figura 11: diagrama entidad relación

5.2.2. Diseño lógico relacional

En esta etapa, se ha transformado el esquema conceptual en un esquema lógico para posteriormente adaptar su estructura de datos al modelo de base de datos en el que se basa nuestro SGBD. La empresa de prácticas, desde el incio del proyecto, ha seleccionado de entre todos los SGBD el sistema gestor MySQL, el cual está basado en un modelo relacional.

La normalización del diseño conceptual es una técnica que se utiliza para comprobar la validez de los esquemas lógicos basados en el modelo relacional, ya que garantiza que las tablas obtenidas no tengan datos redundantes.

A continuación se muestra el resultado de dicha normalización, con el esquema relacional normalizado.

USUARIO(usuario, nombre, password)

CONFIGURACION(id_conf, max_dias, limite_dias, fin_de_semana)

CARACTERISTICA(id_caracteristica, nombre, descripcion) clave alternativa (nombre)

ACTIVIDAD(id_actividad, nombre, descripcion) clave alternativa (nombre)

TIPO_CASA(id_tipo_casa, nombre, descripcion) clave alternativa (nombre)

PROVINCIA(codigo_provincia, nombre) clave alternativa (nombre)

POBLACION(codigo_poblacion, codigo_provincia, nombre) clave alternativa (nombre)

	codigo_provincia		NULOS	BORRADO	MODIFICAR
POBLACION	———————	PROVINCIA	No	Restringir	Propagar

CASA (id_casa, nombre, id_tipo_casa, codigo_poblacion, habitaciones, camas, banyos, personas, base, fin_de_semana, descuento_semana, descuento_mes, activa, publicada)

	id_tipo_casa		NULOS	BORRADO	MODIFICAR
CASA	———————	TIPO_CASA	No	Restringir	Propagar

	codigo_poblacion		NULOS	BORRADO	MODIFICAR
CASA	———————	POBLACION	No	Restringir	Propagar

TEMPORADA (id_temporada, nombre, inicio, fin, dias_reservables, dias_precio)

TEMPORADA_CASA (id_temporada, id_casa, tarifa)

	id_casa		NULOS	BORRADO	MODIFICAR
TEMPORADA_CASA	———————	CASA	No	Propagar	Propagar

			NULOS	BORRA	MODIFI
TEMPORADA_CASA	id_temporada	TEMPORADA	No	Propa	Propa

PERIDO_CIERRE (id_periodo, id_casa, nombre, inicio, fin)

			NULOS	BORRADO	MODIFICAR
PERIDO_CIERRE	id_casa	CASA	No	Propagar	Propagar

RESERVA (id_reserva, id_casa, comentario, total, personas, inicio, fin, estado)

			NULOS	BORRADO	MODIFICAR
PERIDO_CIERRE	id_casa	CASA	No	Propagar	Propagar

CARACTERISTICA_CASA (id_caracteristica, id_casa)

			NULOS	BORRA	MODIFI
CARACTERISTICA_CASA	id_casa	CASA	No	Propa	Propa

			NU	BO	MO
CARACTERISTICA_CASA	id_caracteristica	CARACTERISTICA	No	Pro	Pro

ACTIVIDAD_CASA (id_actividad, id_casa)

			NULOS	BORRADO	MODIFICAR
ACTIVIDAD_CASA	id_casa	CASA	No	Propagar	Propagar

			NULOS	BORRA	MODIFI
ACTIVIDAD_CASA	id_actividad	ACTIVIDAD	No	Propagar	Propagar

5.2.3. Diseño físico e implementación

El diseño físico es el proceso de transformar la descripción de la base de datos a memoria secundaria, determinar las estructuras de almacenamiento y escoger los mecanismos que garanticen un acceso eficiente a los datos.

Como el gestor de base de datos propuesto por la empresa de prácticas es MySQL, se ha decidido no realizar el diseño físico con SQL estándar, sino usar la sintaxis del propio gestor de base de datos.

Grupo de usuarios y sus permisos

Se generan una serie de usuarios con el objetivo de mejorar la seguridad y la consistencia de los datos. Decidimos crear tres usuario: un administrador para gestionar la base de datos que

llamaremos administrador o 'root' y tendrá todos los permisos, un usuario gestor del Backend y otro usuario denominado cliente que estará relacionado con el Frontend.

Se muestra una tabla resumen de los usuarios con sus permisos.

Usuario	Tabla	Permisos
administrador	Todas	Todos
cliente	reserva	Select, insert
	Resto de tablas	Select
gestor	configuracion_reseva	Select, update
	Resto de tablas	Select, insert, delete, update

Cuadro 36: Tabla de usuarios.

Indices

Se ha decidido crear cuatro índices, dos sobre la tabla 'reserva' y otros dos sobre 'periodo_cierre'. La finalidad es agilizar las búsquedas y los filtrados sobre las casas por fechas y periodos. Se considera que se harán un gran número de consultas sobre estos atributos: fecha de inicio y fecha de fin de las respectivas tablas. Por otro lado se ha decidido usar índices de tipo btree, ya que para una casa no hay posibilidad de que las fechas se repitan, es decir, serán valores únicos por cada casa. Los índices btrees son los que el gestor asigna por defecto.

```
CREATE INDEX in_reserva_finicio ON reserva (inicio);

CREATE INDEX in_reserva_ffin ON reserva (fin);

CREATE INDEX in_periodo_cierre_finicio ON periodo_cierre (inicio);

CREATE INDEX in_periodo_cierre_ffin ON periodo_cierre (fin);
```

Sobre los índices existentes a partir del diseño lógico, se ha decidido eliminar los índices generados en la tabla configuracion_reserva, sobre su clave primaria id_conf. El motivo es que la tabla solo tiene un registro, no tiene sentido mantener un índice y una clave primaria. Pero por contra, se ha creado en el campo la restricción de valor único para mantener su integridad.

Reglas de integridad

Hay algunas reglas de integridad en la especificación de requisitos funcionales que no se cumplen con el diseño físico propuesto. Estas reglas se pueden abordar desde dos puntos de vista; desde la capa de aplicación o desde la base de datos mediante trigger. En el proyecto se ha decidido usar triggers sobre todo para aquellas restricciones relacionadas con la integridad de los datos.

A continuación, se muestran las reglas de integridad y los triggers que las mantienen:

Se deben mantener los datos de configuración de la aplicación. Para resolver esta regla se ha decidido que haya un solo registro, el cual debe ser creado por el usuario 'root' del gestor de la db y por contra el usuario gestor solo pueda editarlo.

Sentencia para que id sea único.

```
ALTER TABLE `configuracion_reseva`
        ADD UNIQUE KEY `id_conf` (`id_conf`);
```

Sentencia de inserción generada por el 'root'.

```
INSERT INTO `configuracion_reseva`
        (`max_dias`, `limite_dias`, `fin_de_semana`, `id_conf`)
        VALUES ( NULL, NULL,  `vys`, 1);
```

Este trigger impide que las actualizaciones cambien el id.

```
--
-- Disparadores `configuracion_reserva`
--
DELIMITER $$
CREATE TRIGGER `no_modificar_id`
BEFORE UPDATE ON `configuracion_reserva`
FOR EACH ROW
        IF NEW.id_conf != OLD.id_conf
        THEN
        SET NEW.id_conf = OLD.id_conf;
        END IF
$$
DELIMITER;
```

La aplicación debe tener al menos un usuario administrador. Para resolver este problema, el usuario 'root' creará un usuario temporal para acceder al Backend por primera vez.

```
INSERT INTO `usuario`
        (`usuario`, `nombre`, `password`)
        VALUES
        ('temporal', 'temporal  Apellido1 Apellido2','***');
```

Y posteriormente se ha generado el siguiente trigger para impedir que la aplicación se quede sin usuarios.

```
--
-- Disparadores `usuario`
--
DELIMITER $$
CREATE TRIGGER `al_menos_un_usuario`
BEFORE DELETE ON `usuario`
FOR EACH ROW BEGIN
        DECLARE
                num_de_usuarios INT DEFAULT NULL;
        SELECT count(*) FROM usuario INTO num_de_usuarios;
        IF num_de_usuarios < 2
        THEN
                SIGNAL SQLSTATE '45000'
                SET MESSAGE_TEXT
                = 'La aplicacion no se puede querdar sin usuarios!!';
        END IF;
END
$$
DELIMITER ;
```

Para una misma casa, no se deben solapar los periodos. Es decir, que no se crucen fechas entre periodos de cierre y reservas. Para resolver este requisito, se han creado dos triggers uno en la tabla 'reserva' y otro el la tabla 'perido_cierre'. Se muestra sólo el trigger de reserva ya que en el otro relacionado con los periodos, su sintaxis es practicamente igual.

```
--
-- Disparadores `reserva`
--
DELIMITER $$
CREATE TRIGGER `no_solapamiento_reserva`
BEFORE DELETE ON `reserva`
FOR EACH ROW BEGIN
        DECLARE
                num_de_reserva INT DEFAULT NULL;
        DECLARE
                num_de_cierre INT DEFAULT NULL;
        SELECT count(*) FROM reserva
                WHERE id_reserva=NEW.id_reserva
                AND ((inicio <= NEW.inicio and fin >= NEW.inicio )
                OR ( inicio <= NEW.fin and fin >=NEW.fin))
                INTO num_de_reserva;
        SELECT count(*) FROM periodos_cierre
                WHERE id_reserva=NEW.id_reserva
                AND ((inicio <= NEW.inicio and fin >= NEW.inicio )
                OR ( inicio <= NEW.fin and fin >=NEW.fin))
                INTO num_de_cierre;
        IF (num_de_reserva > 0 ) OR (num_de_reserva > 0 )
        THEN
                SIGNAL SQLSTATE '45000'
                SET MESSAGE_TEXT
                = 'Hay reservas para este perido de tiempo!!';
        END IF;
END
$$
DELIMITER ;
```

5.3. Cliente-Servidor

En este punto se va a ampliar la descripción que se realizó en el punto Sección 2.3 Tecnologías del proyecto. El proyecto consiste en una aplicación web basada en el sistema cliente-servidor y como su nombre indica existens dos capas, la del servidor y la del cliente. Para describir esta sección en un primer momento se habla de la implementación según la capa del servidor y posteriormente según la capa del cliente.

5.3.1. Capa servidor

Como se ha comentado a lo largo la memoria y más concretamente en el punto Subsección 5.2.3 diseño físico, la empresa ha decido usar MySQL [6] como sistema gestor de base de datos. En dicho punto también se aborda la implementación que se ha hecho del gestor. Se describe cómo se ha creado una base de datos única para almacenar la información, además de tres tipos de roles de usuario con sus permisos de acceso a los datos. También se describe cómo se han creado índices adicionales para mejorar el acceso y se han añadido diferentes triggers con el objetivo de mejorar la integridad de los datos.

Para realizar la aplicación web del lado del servidor se ha usado el lenguaje de programación PHP [7], el cual ha sido el encargado de realizar la conexión con la base de datos y de generar el código HTML para su posterior interpretación por la capa de cliente.

- El alumno para la implementación ha intentado seguir un patrón modelo vista controlador separando el código en capas. El el punto Sección 5.4 se hace una descripción más detalla.
- Por otro lado la empresa de prácticas, propuso una serie de directrices para la implementación.

    1. Multi usuario. Esta especificación se resolvió manteniendo los datos del usuario a través de las sesiones que ofrece PHP. Por otro lado la sesión también sirvió de ayuda en otros procesos que requerían almacenar información, procesos con funcionalidades muy similares al uso de un carrito de compra en las tiendas online.
    2. No repetir código en las vistas. Esto que quiere decir que para los elementos comunes a más de una vista deben gestionarse en ficheros diferentes, como es el caso del menú, la cabecera o el pie de página de la vista. Para ello se hizo uso de la sentencia include de PHP que te permite realizar llamadas a otros ficheros. Evidentemente hubo que hacer una clase para gestionar el menú ya que aunque es un elemento común difiere según la vista.
    3. Cada vista solo cargará los ficheros de CSS y JavaScript necesarios. Como la descripción indica, cada vista necesita una serie de archivos CSS y JavaScripts, unos comunes y otros específicos. Cuantos más archivos externos, más tarda en cargar la vista, por tanto se planteó realizar otra clase que gestionaría dichos archivos según en qué vista se muestre.
    4. Archivo de configuración. La empresa de prácticas, especificó que la aplicación se pudiera replicar con facilidad en otros servidores, por tanto se hizo un archivo de

configuración que unificara todas las variables del sistema. En dicho archivo se especifican las variables tales como direcciones del host de la aplicación, el host de la base de datos, el nombre de la bases de datos, los datos de usuarios de la base de datos y ciertas rutas de directorios.

5. Un archivo de log. Dicho archivo debe mantener las acciones que realizan los administradores de la aplicación, sobre todo las acciones de creación, modificación y borrado de registros de la base de datos. Se creó una clase en PHP para gestionar ficheros de log. La gestión consiste en la creación de un fichero al día donde línea a línea se escriben las acciones que los usuarios van realizando.

5.3.2. Capa cliente

Esta capa se refiere a lenguajes y archivos que se ejecutan en browsers modernos.

Se ha hecho uso de los estándares HTML, CSS o JavaScript para ciertos elementos o funcionalidades que con Boostrap, ADMLITE o jQuery no llegaban a ofrecer una solución satisfactoria. Para aprender y consultar se ha hecho uso de los tutoriales que ofrece w3schools [9]. De los 3 estándares se ha encontrado mayor dificultad en el uso del CSS.

- HTML y CSS. Se ha usado para pequeños elementos de visualización de la información que no se contemplan en Bootstrap.
- JavaScript. Usada sobre todo para las validaciones en los formularios y alertas de errores implementando una funcionalidad propia.
- JavaScript. Para el manejo de fechas.
- JavaScript. Para crear estructuras de datos mediante objetos con la finalidad de ayudar a la dinámica del interfaz.

La empresa de prácticas propuso el uso del interfaz ADMLITE 2 el cual usa la tecnología Bootstrap que a su vez usa la libreria jQuery. Para una decripción de las tecnologías se puede acceder a la sección Sección 2.3 de la memoria.

A partir de las interfaces que ofrece ADMLITE se adaptaron a las necesidades del proyecto. Conviene destacar que sobre todo ha sido muy útil en:

- La estructuración general del interfaz como la cabecera, la tabla de estado y el pie.
- En la creación del menú principal.
- La creación de formularios.
- Alertas para el manejo de excepciones.

Cuando las interfaces ADMLITE no ofrecían los elementos necesarios para las necesidades del interfaz, se recurrió a Bootstrap, el cual ayudó a implementar:

- La estructura del contenido de las vistas, ofreciendo un sistema de grid o grilla que se adapta a cualquier tipo de dispositivo ya sea móvil, tablet o pantalla de ordenador de sobremesa.
- El uso de pestañas, para dividir formularios muy grandes.
- El uso de iconos.
- Listados sencillos de pocos elementos.
- Etiquetas de estados, con sus colores predeterminados.
- Notificaciones en el menú.
- Barras de proceso.

La última opción era recurrir a jQuery, cuando Bootstrap no ofecía una funcionalidad adecuada y había que generar una personalizada. Se ha hecho uso de jQuery para:

- El manejo de las pestañas de Bootstrap. Fue necesario rehacer ciertas funcionalidades para que se adaptaran a las validaciones de los formularios.
- Para manejar el DOM, como en dependencias entre inputs, como puede ser un selects que se cargan dinámicamente.
- Para realizar las llamadas AJAX, donde se hacen peticiones asíncronas al servidor.
- Para manejar objetos de tipo JSON.

Para ofrecer un interfaz más atractivo, se ha hecho uso de ciertos plugins libres de terceros. Justamente han sido una de las partes más complejas del proyecto, ya que ha habido que estudiar la documentación de los plugins e incluir en sus eventos código personalizado y así conseguir una funcionalidad adecuada a las necesidades del proyecto.

- FullCalendar [10], desarrollado en jQuery. Calendario complejo, pensado más como una herramienta de agenda.
- Bootstrap-datepicker [11] Desarrollado con Bootstrap y jQuery. Ofrece un interfaz de calendario sencillo que se puede aplicar con los inputs del tipo date.
- DataTables [12] . Desarrollado en jQuery y ofrece un interfaz para listar elementos e incluye muchas funcionalidades tales como un buscador o paginador entre otras.
- Bootstrap-checkbox [13]. Un plugin que ofrece un interfaz atractivo para los combobox usando estilos Bootstrap.
- Charts js [14], librería en JavaScript para generar gráficas.

5.4. Patrón modelo vista controlador

Desde el inicio del proyecto, el alumno se planteó usar el patrón de diseño de software MVC, ya que está familiarizado y ha comprobado su validez en otros proyectos. Por el mismo motivo, el alumno propuso a la empresa de prácticas usar algún framework en PHP que usara dicho patrón. Pero esta propuesta fue desestimada al considerar que un framework implicaría una curvatura de aprendizaje y por tanto un retraso en el proyecto. Aun así, se intentó implementar una solución en PHP basada en el patrón.

A continuación se muestra como se diseñó e implementó el patrón MVC.

1. Una capa que hace de controlador y consiste en una clase que se encarga de la conexión con la base de datos, la vista y el modelo de datos.
2. Otra capa, que se encarga de la lógica y crea la estructura de datos necesaria tanto para la vista como para la base de datos. Esta capa consiste en una serie de clases, con sus seters y geters.
3. Y por último la capa de la vista, donde se presenta la información y incluye el código HTML, CSS, JavaScript.
4. En el siguiente esquema se muestra el ciclo de vida del modelo implementado Figura 12.

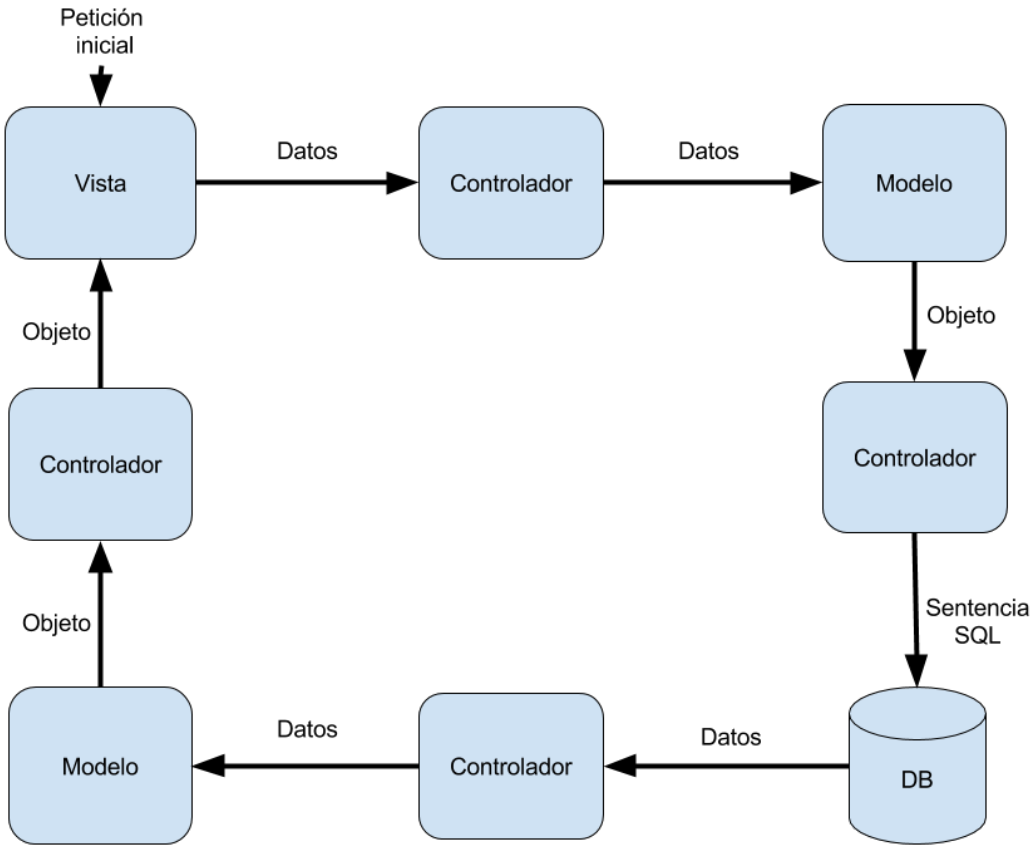

Figura 12: Ciclo de vida del modelo

5.5. Modelo de prototipos

El modelo de prototipos se suele usar en proyectos donde:

- *La mayor dificultad del sistema es identificar de forma correcta y completa los requisitos.* [15]

  Claramente se identifica con la situción en que se encontraba el proyecto después del análisis, donde el cliente no tenía identificados qué datos de entrada y salida eran necesarios para el proyecto.

- *El responsable del desarrollo del software está inseguro de la eficacia de un algoritmo, de la adaptabilidad de un determinado entorno, sistema operativo o de la forma que debería tomar la interacción usuario-sistema.* [15]

  El actual estado del proyecto generaba la incertidumbre de crear una aplicación que cubriera más datos de los necesarios o por el contrario se dejaran datos importantes sin incluir, o ambas situaciones a la vez.

Aunque el modelo de prototipos está pensado para desarrollar sistemas completos y donde el cliente puede ver un sistema en funcionamiento, se planteó sólo desarrollar las interfaces que pertenecen al Backend con su diseño, funcionalidad y su navegación sin la conexión a la base de datos. Es decir usar AdminLITE, HTML, CSS, JavasScript para aportar cierta funcionalidad a las pantallas y su navegación por las mismas. El objetivo era que el cliente identificara los datos necesarios y al mismo tiempo que se sintiera satisfecho con el producto final.

Sin embargo, el modelo de prototipos se basa en la elección de pequeñas porciones de un sistema y se van realizando prototipos, revisiones y mejoras hasta que el cliente está satisfecho. La incorporación de dicho modelo en este punto rompió la planificación inicial del proyecto y a su vez generó la incertidumbre de saber cuánto podía tardar en terminarse su implementación, es decir, no se puede estimar cuánto tiempo implica cada revisión.

1. El primer paso que se planteó fue cómo dividir la aplicación en distintos módulos unitarios y no hubiera una solapación entre ellos, es decir, que desarrollar el prototipo de un módulo no implicara modificar el anterior, tanto a nivel de funcionalidad como de datos. Como resultado se dividió la aplicación en los siguientes módulos, para desarrollarlos de forma independiente:

Procesos	Identificación el sistema Alta Gestor. Modificar gestor. Borrar gestor.
Interfaces	Login. Error login. Nuevo gestor. Listado de gestores. Modificar gestor.

Cuadro 37: Módulo de gestores.

Procesos	Alta de tipo casas. Modificar el tipo casas. Borrar el tipo casas. Listar el tipo casas. Alta de característica. Modificar característica. Borrar característica. Listar características. Alta de actividad. Modificar actividad. Borrar actividad. Listar actividades. Alta de casas. Modificar casa. Borrar casa. Listar casas. Crear período de cierre. Borrar período de cierre.
Interfaces	Listar características. Nueva característica. Modificar característica. Listar actividades. Nueva actividad. Modificar actividad. Listar tipos de casas. Nuevo tipo de casa. Modificar tipo de casa. Listar casas. Nueva casa. Modificar casa. Borrar casa. Ver casa.

Cuadro 38: Módulo de casas.

Procesos	Listar Temporada. Alta temporada. Modificar temporada. Borrar temporada. Asignar tarifa especial por temporada.
Interfaces	Listar temporada. Alta temporada. Modificar temporada. Tarifas por temporada.

Cuadro 39: Módulo de temporadas.

Procesos	Listar reservas. Listar periodos de cierre. Alta reserva. Modificar reserva. Borrar reserva.
Interfaces	Calendario de reservas. Modificar reserva. Nueva reserva.

Cuadro 40: Módulo de reservas.

Procesos	Listar reservas futuras. Listar reservas a pasado.
Interfaces	Estadísticas.

Cuadro 41: Módulo de estadísticas

2. El siguiente paso fue realizar los prototipos de las interfaces con algunos datos ficticios y ciertas funcionalidades en la capa cliente. Como se ha comentado, este paso fue importante para poder recuperar los requisitos de datos necesarios. El responsable analizaba el interfaz y decidía que datos eran necesarios.

3. Por último, con el análisis de datos realizado y la base de datos implementada se empezó a integrar la capa de cliente con la capa del servidor. En este caso el número de revisiones que sufrieron los prototipos fue escasa ya que en el punto anterior se depuraron las funcionalidades.

5.6. Interfaces

5.6.1. Mapa de interfaces

Se ha decidido realizar la Figura 11 para representar el mapa de las pantallas que necesita el sistema. En él se puede apreciar la estructura de interfaces y los recorridos entre ellas, cuáles serán públicas y cuáles privadas además de remarcar en color rojo los caminos iniciales o por defecto.

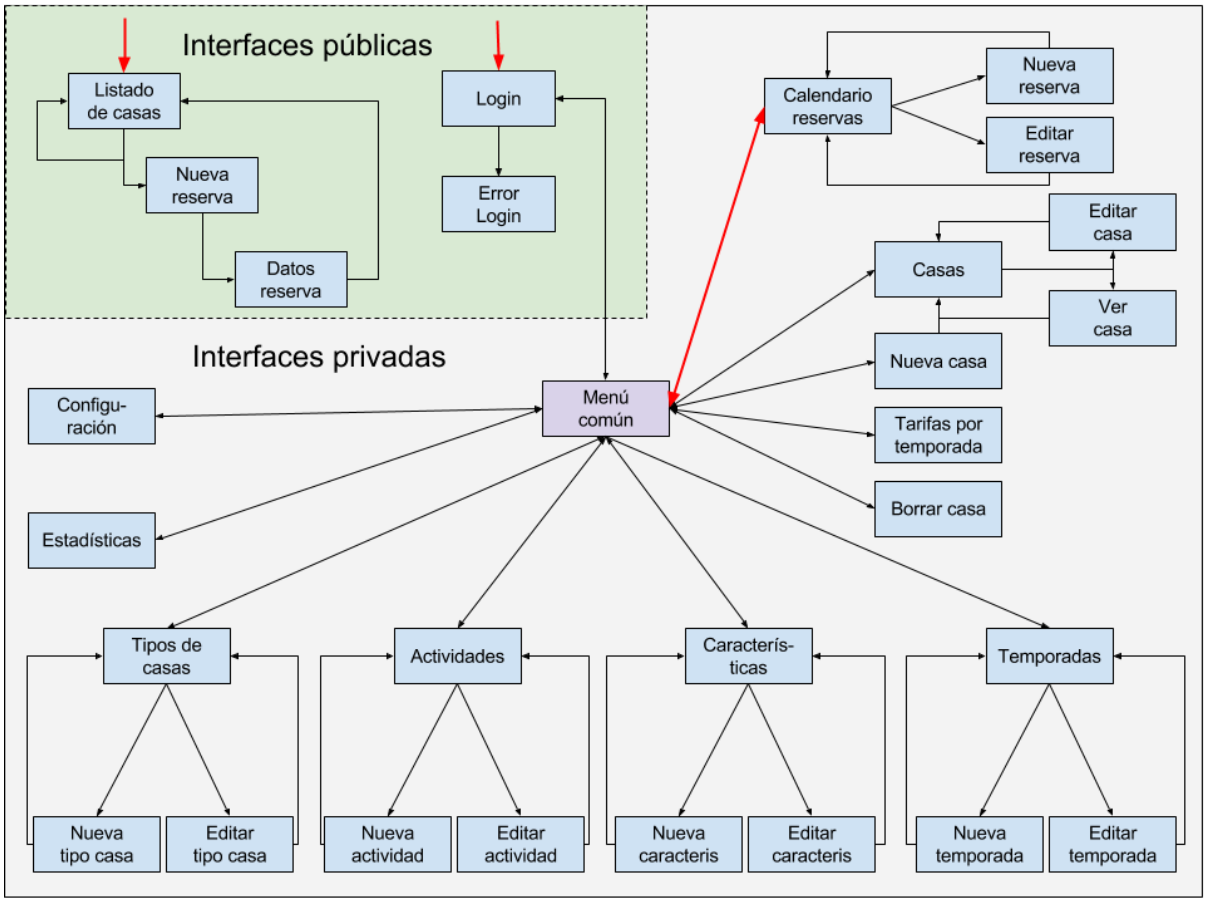

Figura 13: Mapa de interfaces

5.6.2. Ejemplos de interfaces

A continuación se muestran algunas de las interfaces de interés:

- El calendario de reservas. Muestra una lista de las reservas de una forma muy visual e intuitiva pudiendo filtrar por las casas. Figura 14
- El formulario de modificación de una casa. Se ha seleccionado este formulario por la complejidad del interfaz, así como por los elementos de terceros que contiene y por su

funcionalidad. Figura 15, Figura 16 , Figura 17

- El interfaz de estadísticas. Corresponde a un dashboard.Figura 18
- El interfaz tarifas por temporada y casa. Corresponde a una funcionalidad definida por el cliente. Figura 19
- Y por último el interfaz de listado de tarifas. Se ha querido mostrar un listado ya que el sistema está repleto de este tipo de interfaces. Figura 20

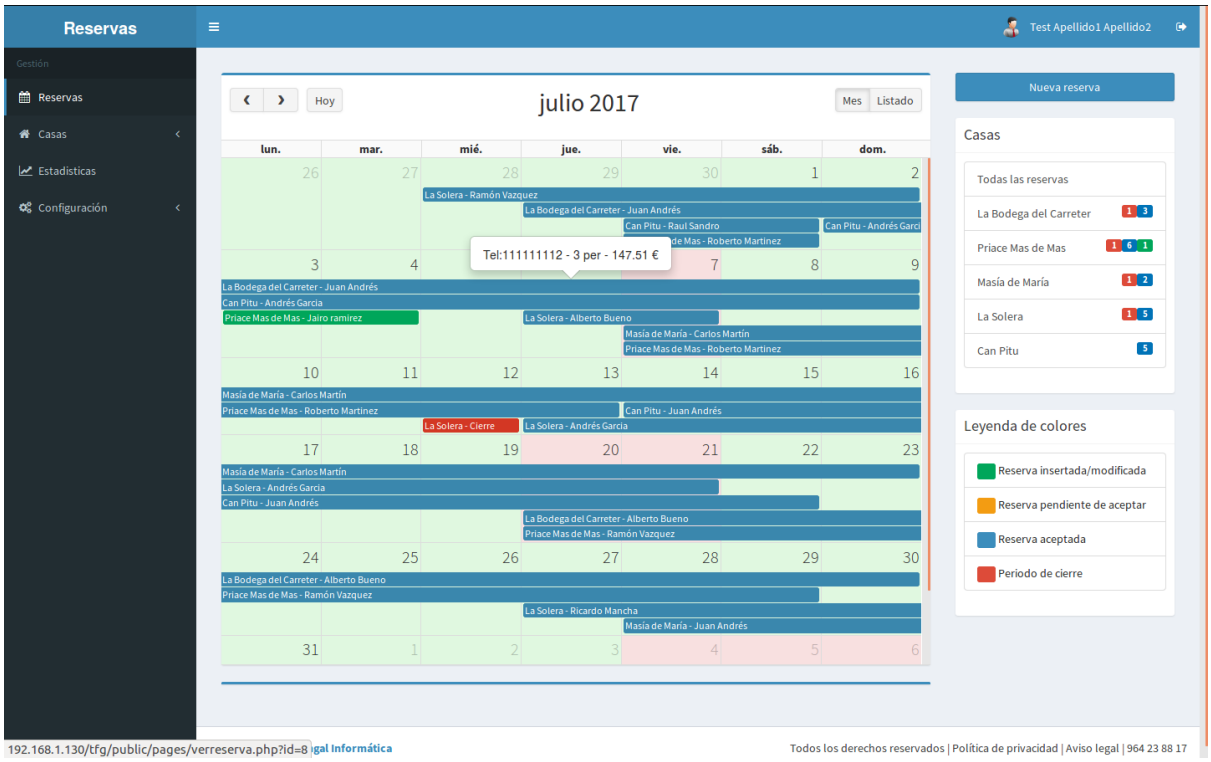

Figura 14: Interfaz de listado de reservas.

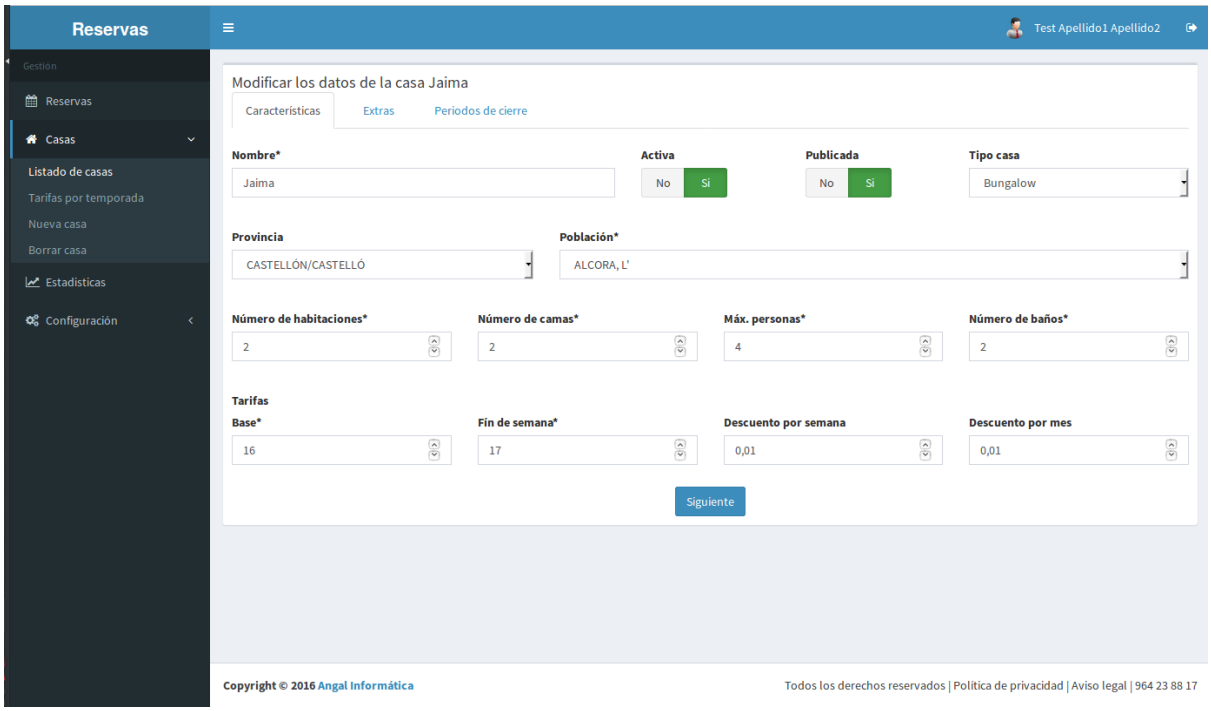

Figura 15: Interfaz de Modificación de casa, paso 1.

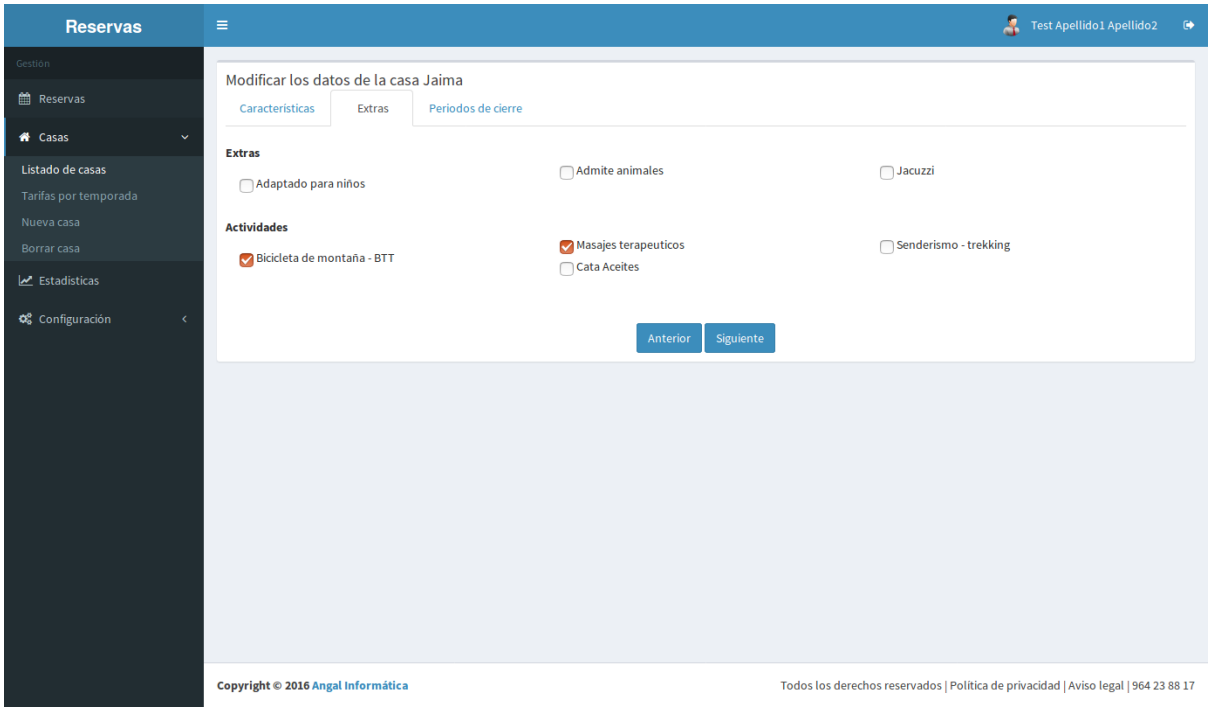

Figura 16: Interfaz de Modificación de casa, paso 2

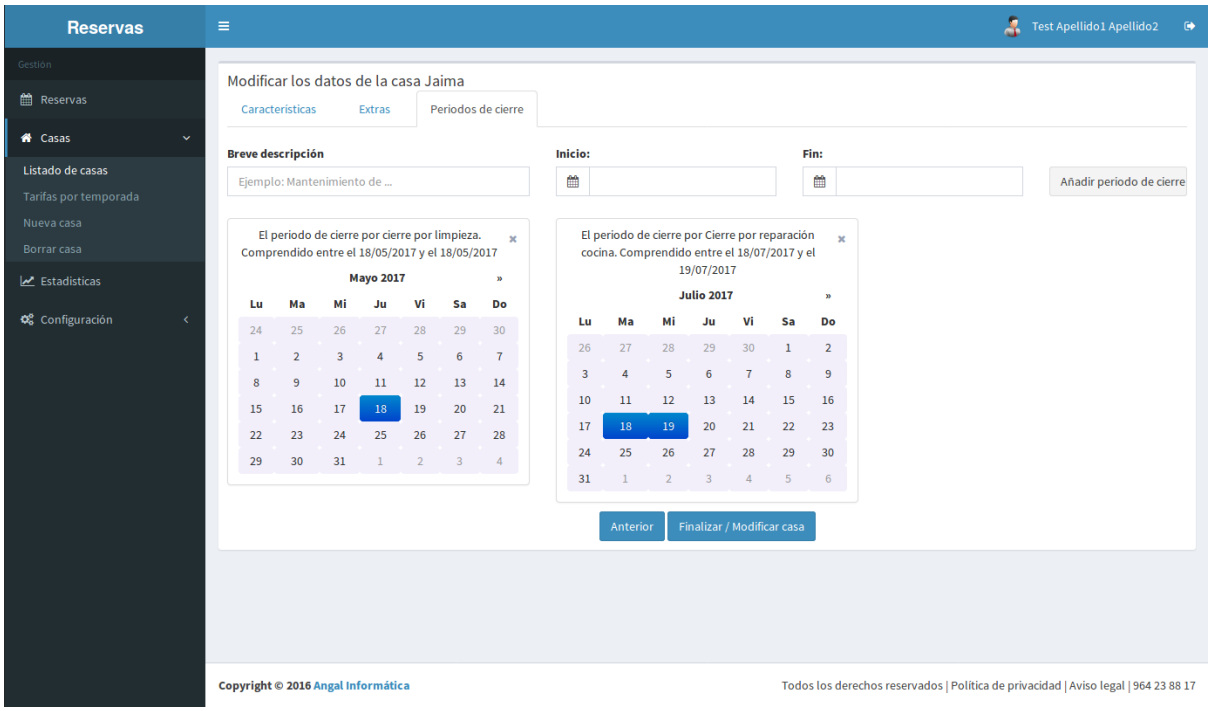

Figura 17: Interfaz de Modificación de casa, paso 3

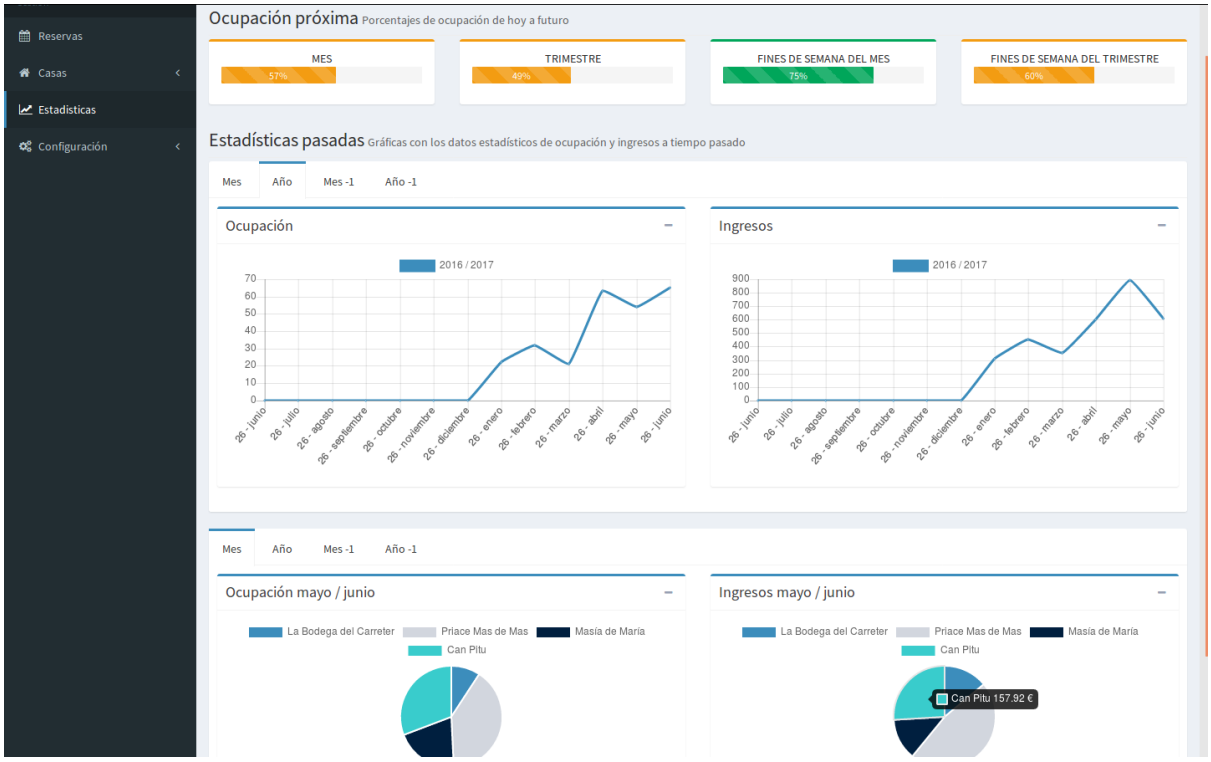

Figura 18: Interfaz de estadísticas

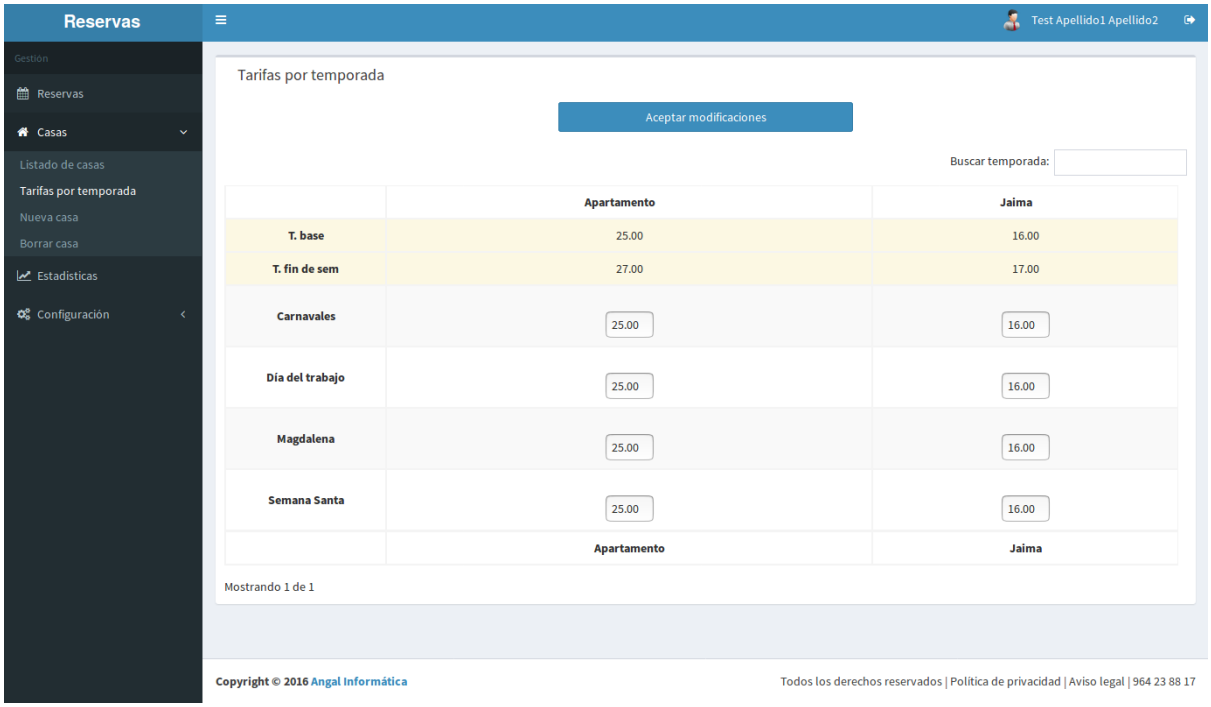

Figura 19: Interfaz de tarifas por casa

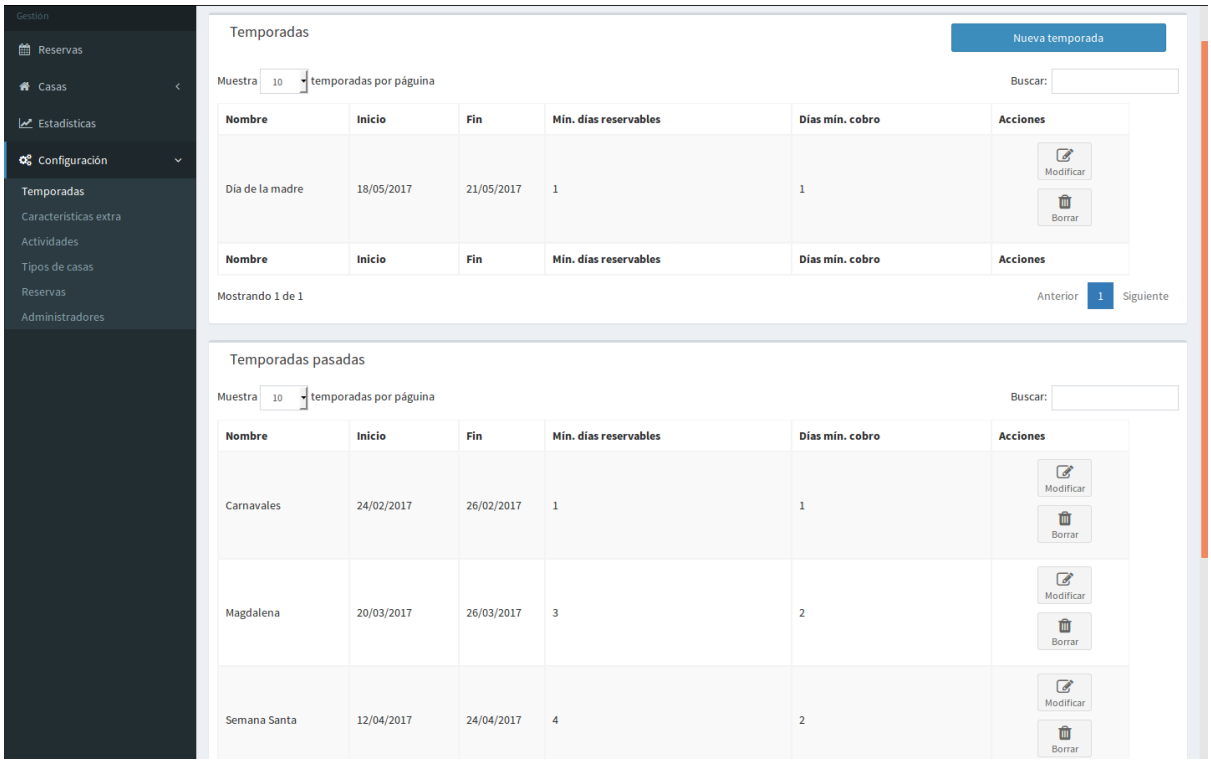

Figura 20: Interfaz de temporadas

5.7. Pruebas

En la empresa no se utiliza ningun tipo de metodología de testing, aun así, se crearon una serie de pruebas con el objetivos de superar las restricciones y validaciones especificadas en el análisis.

A continuación se describen una serie de test superados:

Módulo	Test
Gestores	Login de un usuario gestor. Login de un usuario gestor con datos incorrectos. Acceder al Backend sin pasar por el login. Listado de los todos los gestores Alta de un nuevo gestor datos correctos Alta de un nuevo gestor con el nick de otro gestor Alta de un nuevo gestor con algún dato vacío Modificación de un gestor Modificación gestor con el nick de otro gestor Modificación gestor con algún dato incompleto Borrar un gestor. Borrar el único gestor del sistema. Borrar el gestor autentificado en el sistema .

Cuadro 42: Test de gestores

Módulo	Test
Casas	Alta nuevo de tipo casas. Alta nuevo de tipo casas con el nombre de un tipo ya existente. Alta nuevo de tipo con campos obligatorios vacíos. Modificar un tipo casas. Modificar un tipo casas con el nombre de un tipo ya existente. Modificar un tipo casas con campos obligatorios vacíos. Borrar un tipo casas. Listado de todos los tipos casas. Alta de una nueva característica. Alta de una nueva característica con el nombre de otra característica. Alta de una nueva característica con datos obligatorios vacíos. Modificar característica. Modificar una característica con el nombre de otra caraterística. Modificar una característica con datos obligatorios vacíos. Borrar característica. Listar todos las características. Alta de una nueva actividad. Alta de una nueva actividad con el nombre de otra actividad. Alta de una nueva actividad con datos obligatorios vacíos. Modificar una actividad. Modificar una actividad con el nombre de otra actividad. Modificar una actividad con datos obligatorios vaciós. Borrar actividad. Listar todas las actividades. Alta de una nueva casa con datos correctos. Alta de una nueva casa con con campos obligatorios vacíos. Alta de una nueva casa con con campos con formatos incorrectos. Modificar casa. Modificar una nueva casa con con campos obligatorios vacíos. Modificar una nueva casa con con campos con formatos incorrectos. Borrar casa sin reservas. Borrar casa con reservas. Listar todas las casas. Crear período de cierre correcto. Crear período de cierre con fechas a pasado. Crear período de cierre con fecha inico mayor que la fecha fin. Crear período de cierre con fechas comprendidas en otros periodos. Borrar período de cierre.

Cuadro 43: Test de casas

Módulo	Test
Temporadas	Listar todas las temporadas. Alta temporada correcta. Alta temporada con el nombre de otra temporada. Alta temporada con formato incorrecto de los datos. Alta temporada con datos obligatorios vacíos. Alta temporada con fechas a pasado. Alta temporada con fecha inico mayor que la fecha fin. Alta temporada con fechas comprendidas en otros temporadas. Modificar una temporada. Modificar una temporada con formato incorrecto de los datos. Modificar una temporada con datos obligatorios vacíos. Modificar una temporada con fechas a pasado. Modificar una temporada con fecha inico mayor que la fecha fin. Modificar una temporada con fechas comprendidas en otros periodos. Borrar Temporada. Asignar nuevas tarifas a temporadas por casa. Asignar nuevas tarifas con formatos incorrectos a temporadas por casa.

Cuadro 44: Test de temporadas

Módulo	Test
Reservas	Listar todas las reservas y periodos de cierre a futuro de todas las casas. Listar todas las reservas y periodos de cierre a futuro por casa. Alta una reserva correcta. Alta una reserva con formato incorrecto de los datos. Alta una reserva con datos obligatorios vacíos. Alta una reserva con fechas a pasado. Alta una reserva con fecha inico mayor que la fecha fin. Alta una reserva con fechas comprendidas en otras reservas. Modificar una reserva. Modificar una reserva con formato incorrecto de los datos. Modificar una reserva con datos obligatorios vacíos. Modificar una reserva con fechas a pasado. Modificar una reserva con fecha inico mayor que la fecha fin. Modificar una reserva con fechas comprendidas en otras reservas. Rechazar reserva. Aceptar reserva.

Cuadro 45: Test de reservas

Módulo	Test
Estadísticas	Mostrar datos estadísticos de reservas futuras de forma correcta. Mostrar datos estadísticos de reservas en el pasado de forma correcta.

Cuadro 46: Test de estadísticas

Capítulo 6

Conclusiones

En este apartado final se valora por parte del alumno el trabajo realizado durante la estancia en prácticas y en la elaboración del TFG tanto a nivel técnico como a nivel personal.

El primer punto está relacionado con la valoración por parte del alumno de su participación en un proyecto de naturaleza profesional dentro de un entorno empresarial. El resultado de la estancia en prácticas ha sido en general satisfactorio en cuanto al desarrollo del sistema propuesto por la empresa así como por la vivencia de trabajar dentro de un marco real. Además, el alumno ha podido desarrollar ciertas habilidades tales como:

- Capacidad de aprendizaje.
- Capacidad de afrontar problemas derivados de la implementación.
- Capacidad de comunicación.
- Aportación de ideas.

Las principales dificultades que se han tenido en la implementación han sido sobre todo en el uso de pluggins, ya que ha implicado realizar un estudio de su documentación para que se adaptaran a las funcionalidades requeridas por el sistema. Otro reto fue el manejo de fechas entre los distintos lenguajes del proyecto, ya que cada uno tiene sus propios formatos. Y por último comentar que de entre los tres estándares Web el lenguaje CSS fue la parte más tediosa para obtener los resultados deseados.

El desarrollo y descripción TFG es el otro objetivo que debía superar el alumno, donde ha tenido que aplicar sus conocimientos en Ingeniería del Software y aplicarlos al proyecto. Para ello ha tenido que emplear ciertas capacidades tales como:

- Aplicación los conocimientos adquiridos sobre un proyecto real.
- Planificación y gestión del tiempo.
- Capacidad de análisis y de síntesis.

El resultado en este punto no ha sido del todo satisfactorio ya que el alumno optó inicialmente por una metodología tradicional que no ha sido la adecuada para el proyecto, puesto que se dudó que los requisitos de datos iniciales fueran adecuados o completos. Por otro lado el alumno supo subsanar el problema aplicando el modelo de prototipos en la fase de diseño de las vistas para recuperar los requisitos adecuados. Como consecuencia ha habido un retraso en la planificación y no se ha finalizado la parte Frontend de la aplicación.

Desde el punto de vista de la empresa el resultado no ha sido el esperado, ya que su expectativa era tener el proyecto desarrollado en su totalidad para generar una nueva línea de negocio. Sin embargo, el responsable de la empresa expresó su satisfacción por el Backend desarrollado considerándolo una herramienta muy atractiva y útil para el cliente final. Además manifestó el deseo de poder terminar el proyecto en un futuro.

Posibles ampliaciones y mejoras del proyecto:

- Finalizar el Frontend.
- Banco de pruebas con datos reales.
- Ampliar la funcionalidad del Backend a multipropietario.
- Envío de correos electrónicos.
- Uso de un sistema de versiones.
- Uso de un sistema de pruebas y tests.

Bibliografía

[1] 3WC
https://www.w3.org/standards/webdesign/htmlcss.html Mayo del 2017.

[2] Mozilla Fundation
https://developer.mozilla.org/es/docs/Web/JavaScript Mayo del 2017.

[3] jQuery
https://jquery.com/ Mayo del 2017.

[4] Bootstrap
http://getbootstrap.com/ Mayo del 2017.

[5] The Apache HTTP Server Project
https://httpd.apache.org/ Mayo del 2017.

[6] MySQL
https://www.mysql.com/ Mayo del 2017.

[7] PHP, ¿Qué es PHP?
http://php.net/manual/es/intro-whatis.php Mayo del 2017.

[8] What is UML — Unified Modeling Language
http://www.uml.org/what-is-uml.htm Mayo del 2017.

[9] W3schools Online Web Tutorials
https://www.w3schools.com/ Mayo del 2017.

[10] A JavaScript event calendar. Customizable and open source.
https://fullcalendar.io/ Mayo del 2017.

[11] Bootstrap-datepicker provides a flexible datepicker widget in the Bootstrap style.
https://bootstrap-datepicker.readthedocs.io/en/latest/ Mayo del 2017.

[12] Table plug-in for jQuery
https://datatables.net/ Mayo del 2017.

[13] A checkbox component based on Bootstrap framework.
https://vsn4ik.github.io/bootstrap-checkbox Mayo del 2017.

[14] Chart.js — Open source HTML5 Charts for your website.
http://www.chartjs.org/ Mayo del 2017.

[15] Campos Sancho, Cristina. Grangel Seguer, Reyes. Nebot Romero, María Victoria. 2017. Fonaments d'enginyeria del programari. Publicacions de la Universitat Jaume I.