



**Autores:** Gloria Taboada, Carlos Gómez y Oscar Robles

**Ciclo:** Grado Superior de Desarrollo de Aplicaciones Web (2º curso)

**Clase:** MP12 - Proyecto

**Tutor/a:** Francesca Tomás Artigues



**MÓDULO 13: PROYECTO DE DESARROLLO DE UNA APLICACIÓN MULTIPLATAFORMA**

# Índice

1.1. Propósito del proyecto.....	1
1.2. Objetivos.....	1
1.3. Herramientas utilizadas.....	1
2.1. Recogida de requisitos.....	2
2.2. Análisis.....	4
2.2.1 Diagrama casos de uso.....	4
2.2.2 Especificación de casos de uso.....	5
2.2.1 Diagrama de clases de análisis .....	18
2.3. Diseño.....	19
2.3.1 Diagrama de clases de diseño .....	19
2.3.2 Modelo conceptual y lógico de la base de datos .....	20
2.3.3 Interfaces .....	21
2.3. Codificación y pruebas.....	26
3. Conclusión.....	26
3.1. Objetivos técnicos alcanzados/no alcanzados/ampliados .....	26
3.2. Posibles ampliaciones futuras .....	27
3.3. Valoración personal .....	27
4. Bibliografía.....	28
5. Anexos.....	29
5.1. Plan de Marketing.....	29
5.1.1 Estudio de mercado.....	29
5.1.2 Estrategia de marketing .....	31
5.2. Manual de usuario .....	32
5.3. Manual de instalación.....	39

## 1. Introducció

### 1.1. Propósito del proyecto

Este proyecto nace de la idea de comodidad a la hora de reservar una habitación on-line, ampliando dichas reservas a todos los servicios ofrecidos por el hotel-balneario, tanto de servicios de habitación como de tratamientos de belleza, pasando por reservas de mesa en sus restaurantes o actividades que realice.

La idea es que el cliente del balneario pueda reservar todos los servicios on-line, escogiendo el horario disponible que más le convenga, pudiendo cancelarlo en cualquier momento sin necesidad de llamar por teléfono.

### 1.2. Objetivos

El objetivo del proyecto es facilitar la gestión del personal encargado de hacer las reservas, ahorrando empleados y tiempo al delegar dicha función en el cliente.

El hotel-balneario pondrá a disposición de sus clientes una agenda encargada de mostrar las fechas y horarios disponibles para sus habitaciones, restaurantes y/o tratamientos ofrecidos.

Los clientes tendrán que registrarse en la aplicación para poder gestionar sus reservas.

Como política de www y a fin de que el cliente no pueda acaparar la agenda, se penalizara con el pago de un porcentaje del importe del servicio contratado por reserva anulada sin suficiente anterioridad o según las normas del hotel.

El cliente que no esté registrado y no desee registrarse podrá hacer reservas previo pago, pero no podrá anularlas. De esta manera incentivamos a los clientes a registrarse y facilitarnos sus datos.

### 1.3. Herramientas utilizadas

Los programas utilizados para desarrollar el proyecto han sido los siguientes: Git (GitHub), NetBeans, Composer, MySQL, PhpMyAdmin, Php, Mycrypt, XAMPP y un navegador web (Firefox).

## 2. Desarrollo

### 2.1. Recogida de requisitos

#### Interfaz para el cliente

El proyecto cuenta con una interfaz usable, ágil y de fácil navegación para el cliente. Las diferentes imágenes de cabecera y colores utilizados facilitan la vista del apartado donde se encuentra el cliente.

El menú de la barra de navegación siempre esta visible en cada apartado para una rápida navegación, al igual que el acceso de usuarios registrados o el cambio de idioma.

El diseño de cada apartado está cuidadosamente diseñado para un mayor atractivo a la vista del cliente, usando colores relajantes y diferentes tipos de fuente proporcionadas por *Google Fonts*, que están siempre disponibles para cualquier dispositivo con el que se navegue.

El diseño de la aplicación es totalmente *responsive*, atendiendo a la política *First Mobile*. Las imágenes y las fuentes utilizadas están en medidas escalables, como porcentajes o *vw* (*ancho del viewport*), que permiten ser expuestas en diferentes tamaños sin pérdida de calidad en su representación.

Toda la aplicación está basada en bootstrap 4.

El uso de ventanas modales para mostrar el detalle del producto seleccionado con sus imágenes correspondientes agiliza la carga de cada vista de la aplicación.

La tarea de reserva está simplificada al máximo con sus respectivas notificaciones.

El área personal del cliente registrado está diseñada para que pueda modificar sus datos personales y obtenga un recordatorio de sus anteriores reservas.

#### Interfaz para el administrador

El proyecto cuenta con un menú personalizado para el administrador de la aplicación que le permite ejecutar tareas de administración como la gestión de empleados, clientes y horarios.

Actores y guiones de la aplicación

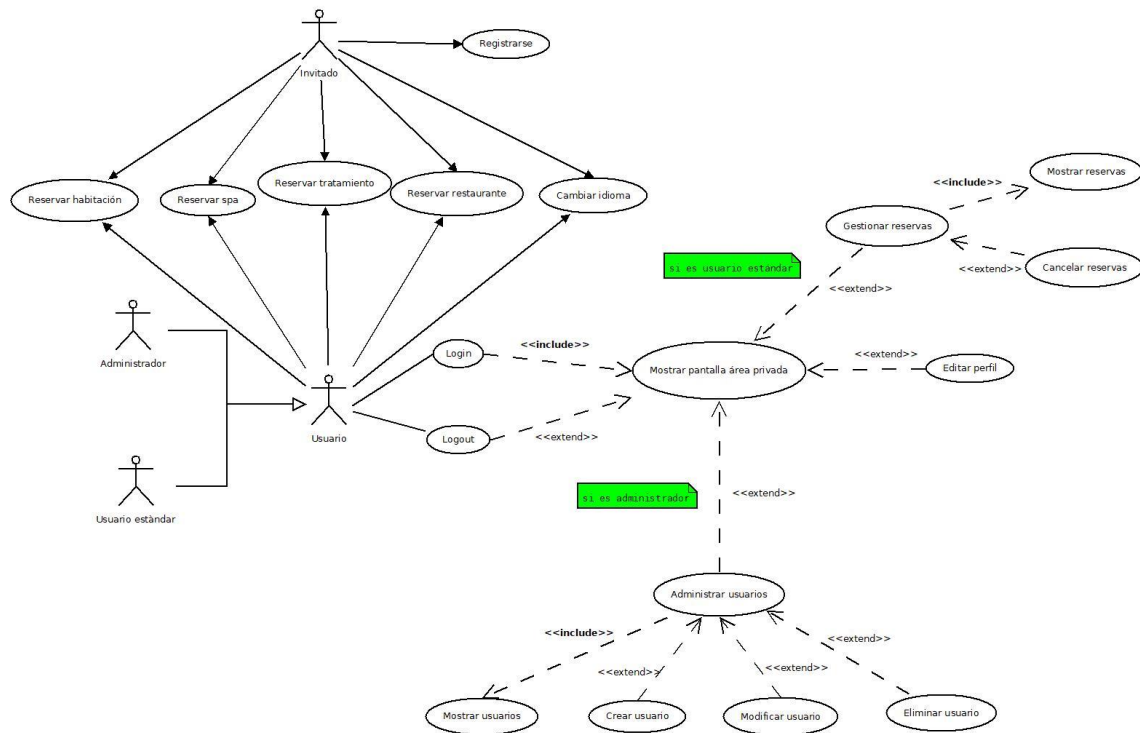
Actor	Administrador
Descripción	Usuario externo que interactúa con la aplicación para administrarla.
Guion	<ul style="list-style-type: none"> <li>• Tiene control total sobre toda la aplicación.</li> <li>• Puede crear, modificar y eliminar cualquier usuario.</li> <li>• Asigna permisos a todos los usuarios.</li> <li>• Puede gestionar reservas.</li> <li>• Puede modificar horarios.</li> </ul>

Actor	Usuario
Descripción	Usuario externo que interactúa con la aplicación.
Guion	<ul style="list-style-type: none"> <li>• Puede mirar toda la información de la aplicación que no requiera permisos de administrador.</li> <li>• Puede registrarse en la aplicación.</li> <li>• Puede modificar sus datos de perfil.</li> <li>• Puede hacer y gestionar sus reservas.</li> <li>• Puede anular reservas.</li> </ul>

Actor	Visitante
Descripción	Usuario externo que interactúa con la aplicación.
Guion	<ul style="list-style-type: none"> <li>• Puede mirar toda la información de la aplicación que no requiera permisos adicionales.</li> </ul>

## 2.2. Anàlisis

### 2.2.1 Diagrama casos de uso



## 2.2.2 Especificación de casos de uso

Nombre: Registrarse	
Descripción: Acceder a la aplicación	
Actores: Visitante	
PRECONDICIÓN: Nada	
Flujo Normal	Flujo Alternativo
1. El usuario selecciona la opción para acceder al área de usuarios.	
2. Clica en el botón para registrarse y el sistema le muestra una pantalla de formulario donde ha de incluir los datos del registro.	
3. El sistema verifica los datos.	3.1 El sistema indica que el usuario ya está registrado mostrándole la pantalla del formulario para que vuelva a insertar los datos de registro 3.2. El sistema verifica los datos. 3.3 Se produce un error. Avisa al visitante que se ha producido un error.
5. El sistema registra al visitante	
POSTCONDICIÓN: El usuario visitante se ha registrado.	

Nombre: Login	
Descripción: Acceder al área privada de la aplicación	
Actores: Usuario (estándar y Administrador)	
PRECONDICIÓN: Estar registrado en el sistema	
Flujo Normal	Flujo Alternativo
1. El usuario selecciona la opción para hacer login.	
2. El sistema pide nombre de usuario y contraseña.	
3. El usuario introduce los datos de usuario y contraseña, clic en aceptar.	3.1 El usuario introduce parcialmente los datos y clic en aceptar. El sistema avisa que completes el login y vuelve al punto 1. 3.2 No introduce ningún dato y clic en aceptar. El sistema avisa que completes el login y vuelve al punto 1. 3.3 El usuario no hace nada.
4. El sistema valida los datos introducidos.	4.1 El sistema no encuentra el usuario en la BBDD. El sistema avisa de que no existe el usuario y vuelve al punto 1. 4.2 El sistema encuentra el usuario, pero no coincide la contraseña. El sistema avisa de que la contraseña no es válida.
5. El usuario entra en el sistema	
POSTCONDICIÓN: El usuario ha podido acceder al sistema y muestra su área privada	



Nombre: Mostrar pantalla principal	
Descripción: Muestra la interfaz del área privada con las funcionalidades principales	
Actores: Usuario (estándar y Administrador)	
PRECONDICIÓN: Estar logueado en el sistema	
Flujo Normal	Flujo Alternativo
<p>1.1. El sistema muestra la interfaz del área privada y habilita las siguientes funcionalidades:</p> <p>Si es administrador muestra:</p> <ul style="list-style-type: none"> <li>· Editar perfil</li> <li>· Ver reservas</li> <li>· Logout</li> </ul> <p>Si es administrador muestra:</p> <ul style="list-style-type: none"> <li>· Editar perfil</li> <li>· Administrar usuarios</li> <li>· Logout</li> </ul>	
POSTCONDICIÓN: El sistema muestra la interfaz del área privada	

Nombre: Logout	
Descripción: Hacer Logout y abandonar el área privada	
Actores: Usuario estándar y Administrador	
PRECONDICIÓN: Estar logueado	
Flujo Normal	Flujo Alternativo
1.El usuario selecciona la opción de logout	
POSTCONDICIÓN: El usuario accede a la página principal de la web	

Nombre: Editar Perfil	
Descripción: El usuario puede cambiar sus datos personales	
Actores: Usuario estándar y Administrador	
PRECONDICIÓN: Estar logueado	
Flujo Normal	Flujo Alternativo
1. El usuario selecciona la opción Editar Perfil	
2.El sistema muestra los datos de usuario en modo edición: <ul style="list-style-type: none"> <li>Email</li> <li>Nombre</li> <li>Numero tarjeta</li> </ul>	
3. El usuario modifica los datos y clics en guardar.	3.El usuario no modifica los datos
4.El sistema valida las modificaciones y vuelve a la página principal de usuario	4.1. El usuario no ha modificado ningún dato. Vuelve a la página principal de usuario. 4.2. El sistema encuentra no válidas las modificaciones.
POSTCONDICIÓN: El sistema registra las modificaciones y vuelve a la página principal de usuario.	

Nombre: Mostrar reservas	
Descripción: El usuario puede ver sus reservas	
Actores: Usuario estándar	
PRECONDICIÓN: Estar logueado	
Flujo Normal	Flujo Alternativo
1. El sistema busca las reservas que tiene el usuario	
2. El sistema devuelve las reservas seleccionadas del punto anterior.	2. El sistema no devuelve nada por que el usuario no tiene reservas
POSTCONDICIÓN: Devuelve todas las reservas.	

Nombre: Gestionar reservas	
Descripción: El administrador puede ver las reservas realizadas	
Actores: Usuario	
PRECONDICIÓN: Estar logueado	
Flujo Normal	Flujo Alternativo
1.El sistema busca las reservas que tiene el usuario en la BBDD	
2. El sistema devuelve las reservas seleccionadas del punto anterior.	2. El sistema no devuelve nada porque el usuario no tiene reservas
3. Si el usuario selecciona una reserva y desea cancelarla se ejecuta "Cancelar reserva"	
POSTCONDICIÓN: Devuelve todas las reservas.	

Nombre: Cancelar reservas	
Descripción: El usuario cancela reservas	
Actores: Usuario estándar	
PRECONDICIÓN: Estar logueado	
Flujo Normal	Flujo Alternativo
1.El sistema pregunta si desea eliminar una reserva	
2. El usuario selecciona sí	2. El usuario selecciona no 2.1 Se vuelve a la pantalla de reservas
3.Se cancela la reserva	
POSTCONDICIÓN: Devuelve todas las reservas.	

Nombre: Mostrar usuarios	
Descripción: Mostrar usuarios del sistema	
Actores: Administrador	
PRECONDICIÓN: Estar logueado	
Flujo Normal	Flujo Alternativo
1. El sistema busca en la bbdd todos los usuarios registrados	
POSTCONDICIÓN: El sistema devuelve todos los usuarios	

Nombre: Administrar usuarios	
Descripción: El administrador puede ver los usuarios registrados	
Actores: Administrador	
PRECONDICIÓN: Estar logueado	
Flujo Normal	Flujo Alternativo
1.El sistema busca los usuarios que tiene el sistema en la BBDD	
2. El sistema devuelve los usuarios seleccionadas del punto anterior.	2. El sistema no devuelve nada porque el sistema no tiene usuarios registrados
3.1 Si el administrador selecciona un usuario y desea eliminarlo se ejecuta “Eliminar usuario”  3.2 Si el administrador selecciona un usuario y desea modificarlo se ejecuta “Modificar usuario”  3.3 Si el administrador desea crear un usuario se ejecuta “Crear usuario”	
POSTCONDICIÓN: Devuelve todos los usuarios.	

Nombre: Eliminar usuario	
Descripción: El administrador elimina usuarios	
Actores: Administrador	
PRECONDICIÓN: Estar logueado	
Flujo Normal	Flujo Alternativo
1. El sistema pregunta si desea eliminar ese usuario	
2. El usuario selecciona “sí”	2. El usuario selecciona “no” 2.1 Se vuelve a la pantalla de reservas
3. Se elimina el usuario	
POSTCONDICIÓN: El usuario se ha eliminado y devuelve todos los usuarios.	

Nombre: Crear Usuario	
Descripción: El administrador da de alta usuarios	
Actores: Administrador	
PRECONDICIÓN: Estar logueado	
Flujo Normal	Flujo Alternativo
1. El usuario selecciona la opción Crear Usuario	
2. El sistema muestra los datos de usuario en modo edición: <ul style="list-style-type: none"> <li>Email</li> <li>Nombre</li> </ul>	
3. El sistema verifica los datos.	3.1 El sistema indica que el usuario ya está registrado
5. El sistema crea el usuario	
POSTCONDICIÓN: El usuario se crea	

Nombre: Modificar Usuario	
Descripción: El administrador modifica usuarios	
Actores: Administrador	
PRECONDICIÓN: Estar logueado	
Flujo Normal	Flujo Alternativo
1. El usuario selecciona la opción Modificar Usuario	
2.El sistema muestra los datos de usuario en modo edición: <ul style="list-style-type: none"> <li>Email</li> <li>Nombre</li> </ul>	
3. El administrador modifica los datos y clicka en guardar	3.1 El administrador no modifica los datos
4.El sistema valida las modificaciones y vuelve a la página principal de administrador	4.1. El administrador no ha modificado ningún dato. Vuelve a la página principal de administrador 4.2. El sistema encuentra no válidas las modificaciones.
POSTCONDICIÓN: El usuario se modifica	

Nombre: Cambiar idioma	
Descripción: Cambiar idioma de la aplicación	
Actores: Usuario (estándar y Administrador) y visitante	
PRECONDICIÓN: NADA	
Flujo Normal	Flujo Alternativo
1. El usuario selecciona la opción per cambiar el idioma de la web.	
POSTCONDICIÓN: La página se ha cambiado de idioma.	

Nombre: Reservar Habitación	
Descripción: Reservar una habitación	
Actores: Usuario (estándar y Administrador) y visitante	
PRECONDICIÓN: Nada	
Flujo Normal	Flujo Alternativo
1. El usuario logueado o visitante selecciona la opción de hacer una reserva de habitación.	
2. El sistema muestra los datos de la reserva de la habitación <ul style="list-style-type: none"> <li>· Nombre habitación</li> <li>· Fecha llegada</li> <li>· Fecha salida</li> <li>· Huéspedes</li> </ul> Si es invitado también muestra: <ul style="list-style-type: none"> <li>· Número de tarjeta</li> </ul>	
3. El usuario introduce los datos y clicla en “reservar”	3.1 El usuario no introduce datos y clicla en “volver”.
3. El sistema verifica los datos	3.1 El sistema indica que los datos introducidos no son correctos
4. El sistema registra la reserva	
POSTCONDICIÓN: El usuario hace la reserva y vuelve a la página de habitaciones.	

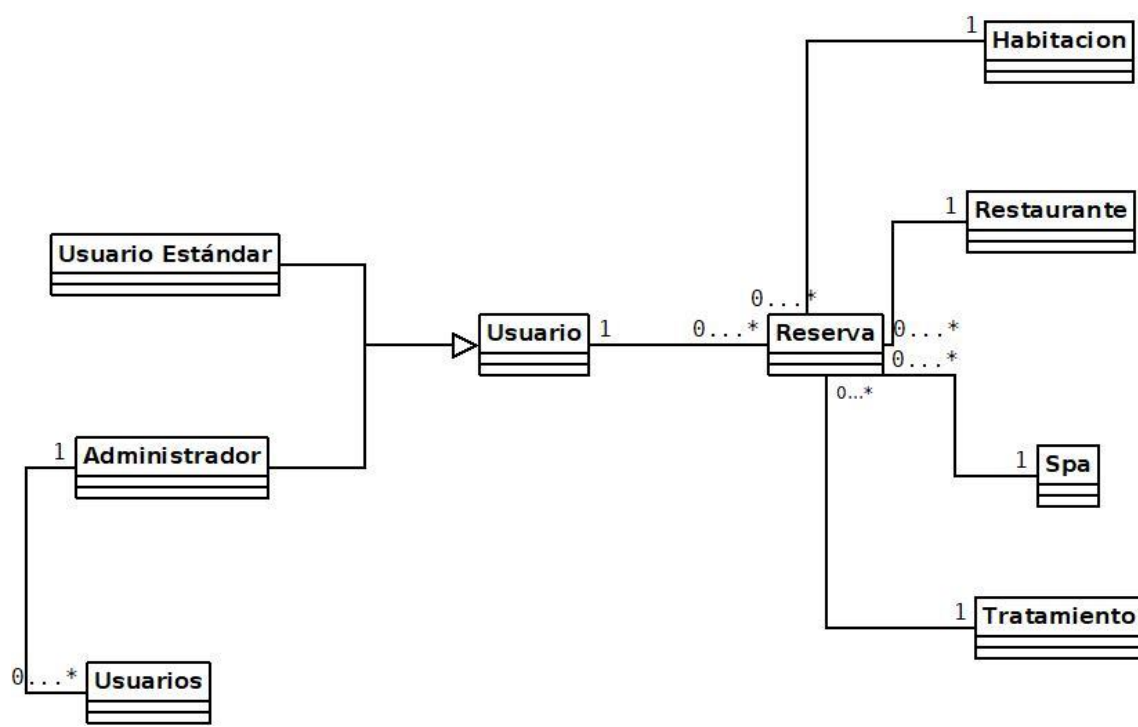


Nombre: Reservar Restaurante	
Descripción: Reserva en un restaurante	
Actores: Usuario (estándar y Administrador) y visitante	
PRECONDICIÓN: Nada	
Flujo Normal	Flujo Alternativo
1. El usuario logueado o visitante selecciona la opción de hacer una reserva de restaurante.	
2. El sistema muestra los datos de la reserva del restaurante <ul style="list-style-type: none"> <li>· Nombre restaurante</li> <li>· Fecha</li> <li>· Comensales</li> <li>· Hora</li> </ul> Si es invitado también muestra: <ul style="list-style-type: none"> <li>· Número de tarjeta</li> </ul>	
3. El usuario introduce los datos y clicla en "reservar"	3.1 El usuario no introduce datos y clicla en "volver".
3. El sistema verifica los datos	3.1 El sistema indica que los datos introducidos no son correctos
4. El sistema registra la reserva	
POSTCONDICIÓN: El usuario hace la reserva y vuelve a la página de restaurantes.	

Nombre: Reservar Balneario-Spa	
Descripción: Reserva en un balneario-spa	
Actores: Usuario (estándar y Administrador) y visitante	
PRECONDICIÓN: Nada	
Flujo Normal	Flujo Alternativo
1. El usuario logueado o visitante selecciona la opción de hacer una reserva de un balneario-spa.	
2. El sistema muestra los datos de la reserva de la balneario-spa <ul style="list-style-type: none"> <li>· Nombre balneario-spa</li> <li>· Fecha</li> <li>· Personas</li> <li>· Hora</li> </ul> Si es invitado también muestra: <ul style="list-style-type: none"> <li>· Número de tarjeta</li> </ul>	
3. El usuario introduce los datos y clicla en “reservar”	3.1 El usuario no introduce datos y clicla en “volver”.
3. El sistema verifica los datos	3.1 El sistema indica que los datos introducidos no son correctos
4. El sistema registra la reserva	
POSTCONDICIÓN: El usuario hace la reserva y vuelve a la página de balneario-spa	

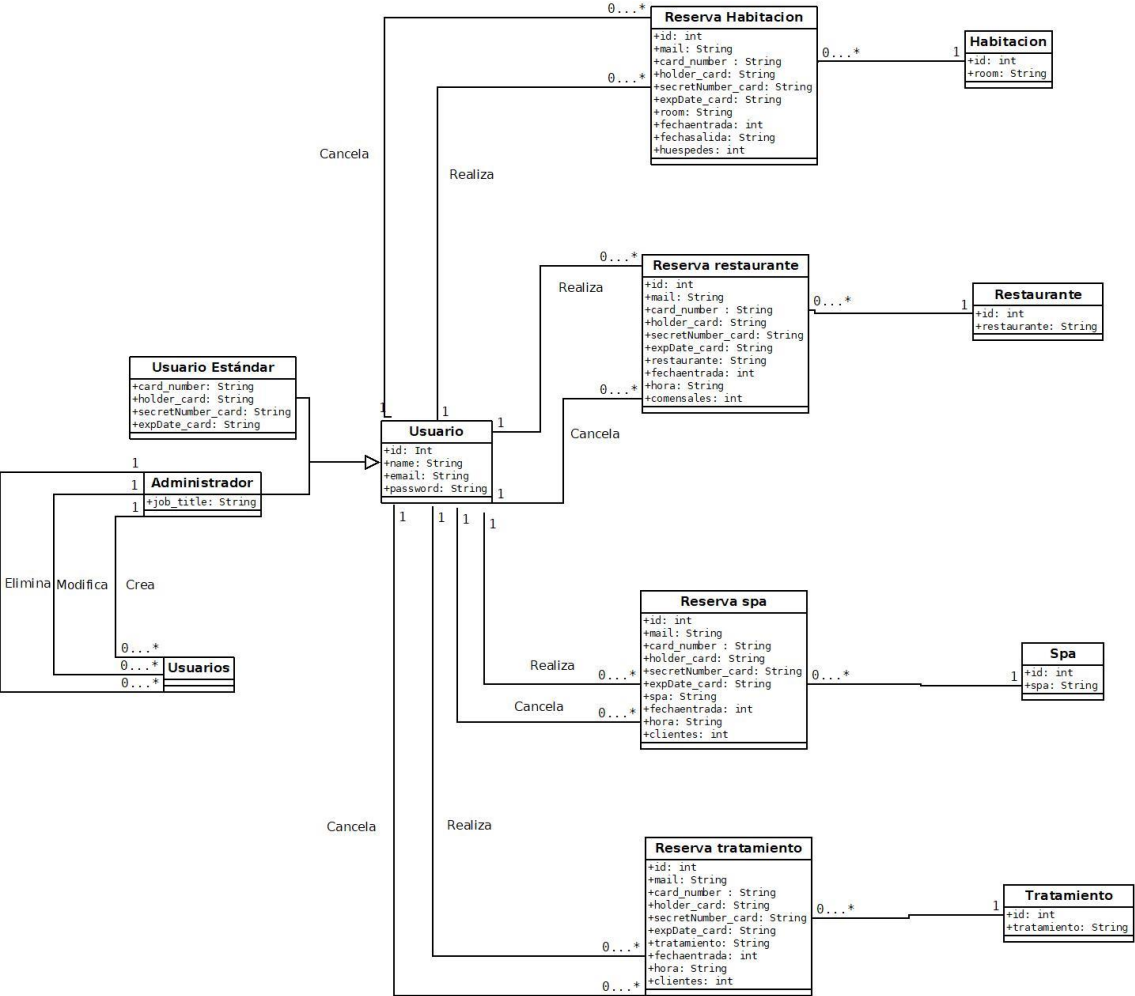
Nombre: Reservar Tratamiento	
Descripción: Reserva un tratamiento	
Actores: Usuario (estándar y Administrador) y visitante	
PRECONDICIÓN: Nada	
Flujo Normal	Flujo Alternativo
1. El usuario logueado o visitante selecciona la opción de hacer una reserva de un tratamiento.	
2. El sistema muestra los datos de la reserva de un tratamiento <ul style="list-style-type: none"> <li>· Nombre tratamiento</li> <li>· Fecha</li> <li>· Hora</li> </ul> Si es invitado también muestra: <ul style="list-style-type: none"> <li>· Número de tarjeta</li> </ul>	
3. El usuario introduce los datos y clicla en “reservar”	3.1 El usuario no introduce datos y clicla en “volver”.
3. El sistema verifica los datos	3.1 El sistema indica que los datos introducidos no son correctos
4. El sistema registra la reserva	
POSTCONDICIÓN: El usuario hace la reserva y vuelve a la página de tratamientos	

## 2.2.1 Diagrama de classes de anàlisis

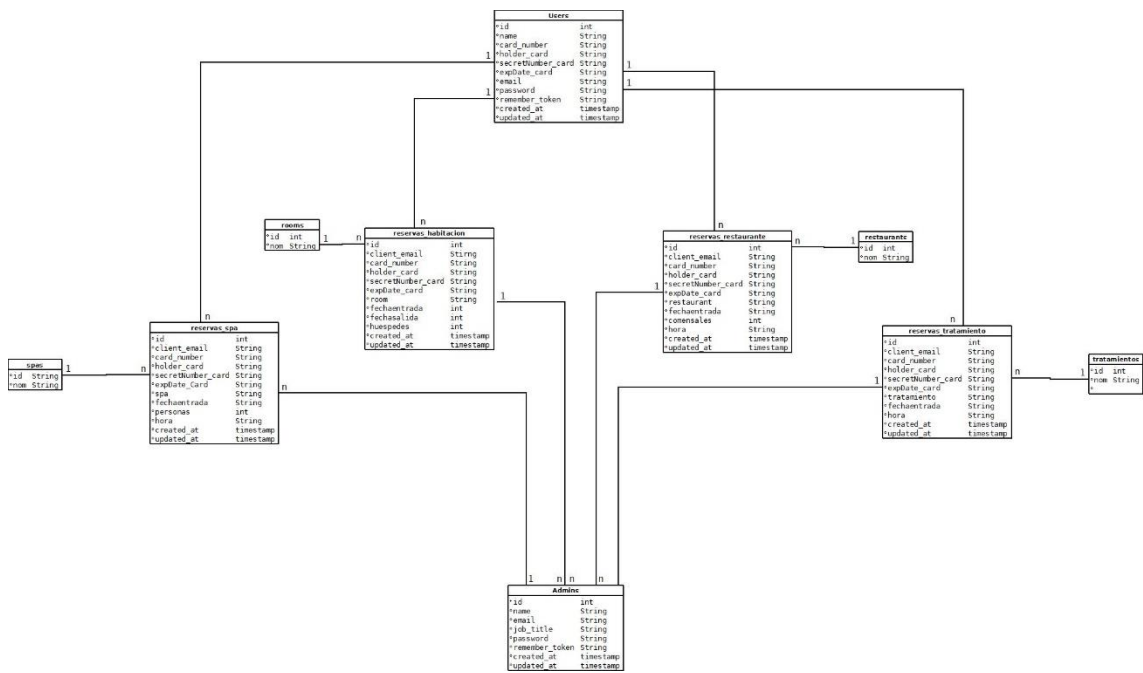


2.3. Diseño

2.3.1 Diagrama de clases de diseño

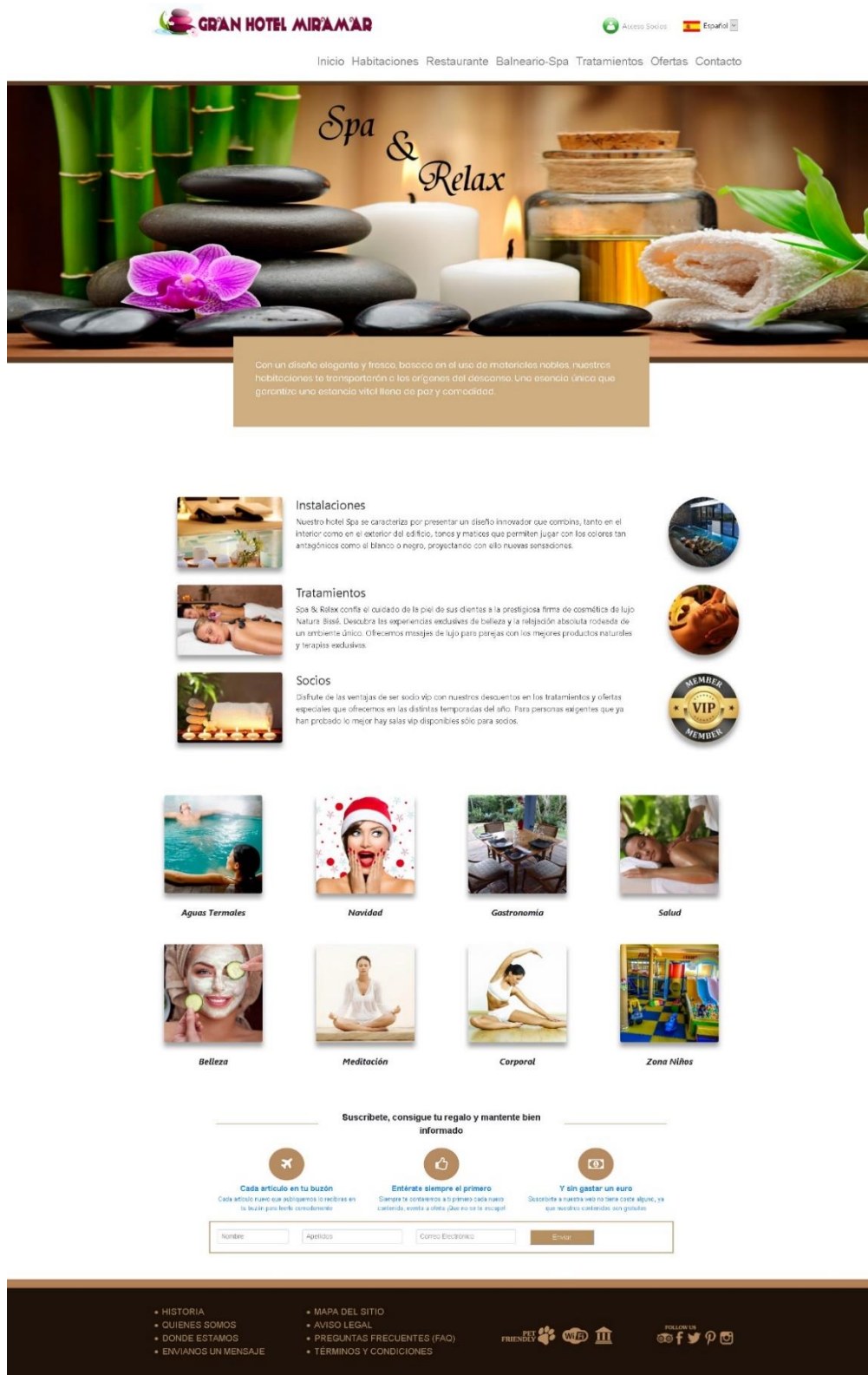


2.3.2 Modelo conceptual y lógico de la base de datos



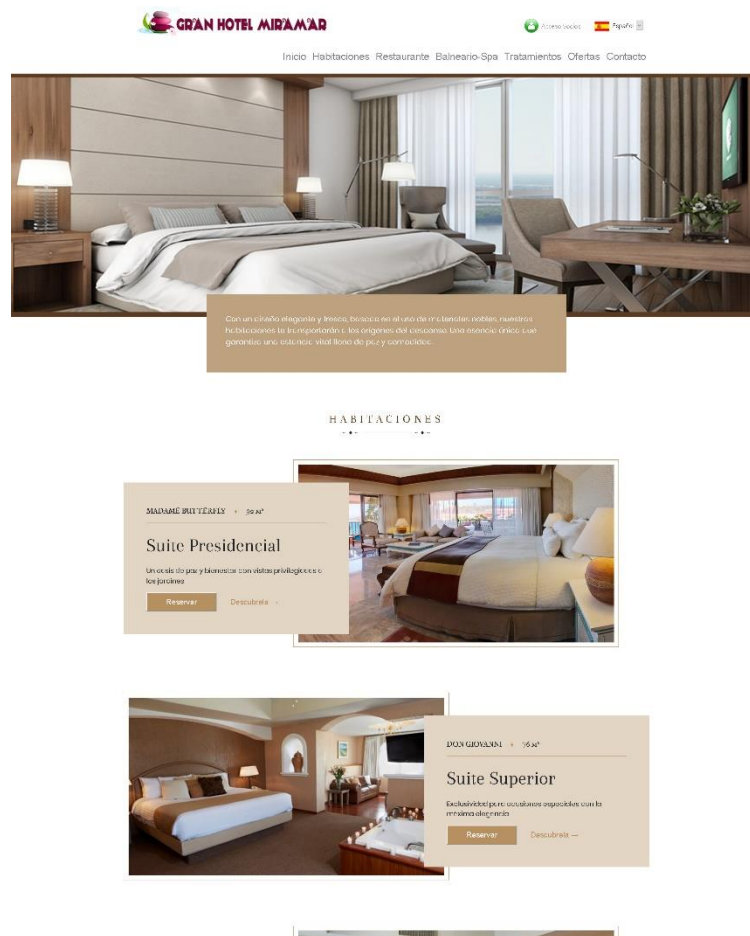
## 2.3.3 Interfaces

### -Página Principal



# INS “Nicolau Copèrnic” Desenvolupament d'Aplicacions Multiplataforma

## -Página Habitaciones (resto de secciones, misma estructura)



## -Página Contacto



## -Inicio de sesión usuario



[Admin login](#) [User Login](#) [Register](#)  [English](#) 

Login

E-Mail Address

Password

☐ Remember Me

Login

[Forgot Your Password?](#)

## -Registro de usuario



[Admin login](#) [User Login](#) [Register](#)  [English](#) 

Register

Name

E-Mail Address

Targeta de pago

Titular tarjeta

Fecha cad. tarjeta

Num. secreto tarjeta

Password

Confirm Password

Register

## -Recuperación de contraseña



[Admin login](#) [Login](#) [Register](#)  [English](#) 

Reset Password

E-Mail Address

Send Password Reset Link

## -Panel perfil de usuario



## -Vista “Mis datos” (Usuario)

The screenshot shows a modal form titled 'Mis datos' (My Data) with a close button (X). The form contains the following fields:

- Nombre: Prueba
- Correo electrónico: prueba@prueba.com
- Tarjeta de pago: 1234567890123455
- Titular tarjeta: Oscar Roblaaaaaas
- Fecha cad. tarjeta: 02/22
- Num. secreto tarjeta: ...

At the bottom of the form are two buttons: 'Guardar' (Save) and 'Cerrar' (Close).

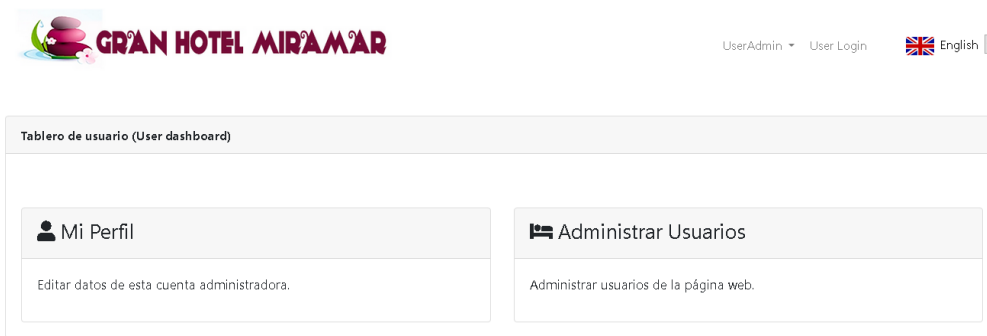
## -Vista “Mis reservas” (Usuario)

The screenshot shows a modal form titled 'Mis reservas' (My Reservations) with a close button (X). It contains a table with the following data:

Tipo reserva	Info reserva	Fecha entrada	Fecha salida   Hora	Opciones
Restaurante	Spring Space	01/06/2018	13:00	Cancelar reserva

At the bottom of the modal is a 'Cerrar' (Close) button.

## -Panel perfil de administrador



## -Vista “Mis datos” (Administrador)

The 'Mis datos' form contains the following fields:

- Nombre: UserAdmin
- Correo electrónico: admin@hotelspa.com
- Trabajo: Administrador\_BBDD

Buttons: Guardar, Cerrar

## -Vista “Administrar Usuarios” (Administrador)

Nombre	Correo electrónico	Número tarjeta	Titular tarjeta	Número secreto tarjeta	Fecha caducidad tarjeta	
no	k@k.com	1234567890123456				Eliminar usuario
Prueba	prueba@prueba.com	1234567890123455	Oscar Robblaaaaaas	556	02/22	Eliminar usuario

Buttons: Guardar, Cerrar

## 2.3. Codificación y pruebas

La codificación y las pruebas de la aplicación se han llevado a cabo entre los tres miembros del grupo. Una vez que las pruebas se realizaban con éxito el código utilizado se unía con el resto del proyecto a través de la plataforma GitHub.

## 3. Conclusión

### 3.1. Objetivos técnicos alcanzados/no alcanzados/ampliados

#### Objetivos técnicos alcanzados

Los objetivos alcanzados son haber empezado un proyecto desde cero, planificando cada apartado y basándonos en un diseño creado a partir de una imagen adjuntada en la propuesta de proyecto, clonándola en formato web con la ayuda de css y bootstrap 4, herramienta que hemos aprendido a usar por nuestra cuenta con espectaculares resultados, ya que durante el curso hemos utilizado la versión 3.

El sistema de autenticación diferencia entre logueo de usuarios y logueo de administrador, teniendo este último todas las herramientas para crear, editar y eliminar usuarios en ventanas modales bajo petición Ajax y así optimizar la carga de información sin requerir muchos recursos.

La aplicación está diseñada para abarcar varios idiomas.

Se ha logrado con todo lo desarrollado crear una página web de reservas de un hotel para usuarios e incluso para invitados para realizar reservas si la necesidad de estar registrados en la aplicación.

#### Objetivos técnicos no alcanzados

Como objetivo técnico no alcanzado sería la implementación de un calendario donde se reflejarán los días que ya están ocupados.

No obstante, es un objetivo no alcanzado por falta de tiempo y no por falta de conocimientos.

#### Objetivos técnicos ampliados

Un objetivo técnico ampliado a tener en cuenta es el sistema empleado para traducir el contenido de la página, que facilita mucho la tarea, pudiendo traducir a cualquier idioma el contenido de un apartado solo con añadir un nuevo archivo con la traducción del texto deseado.

### 3.2. Posibles ampliaciones futuras

- Una posible ampliación futura sería añadir los idiomas más usados habitualmente.
- Otra posible ampliación sería enviar un email al usuario que quisiera reservar habitación u otro servicio no habiendo estado está disponible para las fechas seleccionadas, que previamente habría cancelado otro usuario.
- La posibilidad de ponerse en contacto por medio del envío del formulario de contacto a través de correo electrónico con el administrador de la aplicación.
- La importación de otro acceso a reservas donde se puedan ver todos los tipos de reservas del hotel (habitaciones, restaurante, spa-balneario, y tratamientos) en una misma vista.
- Encriptación de datos bancarios.

### 3.3. Valoración personal

*Oscar*

En general este proyecto me ha servido como un trabajo para poner en práctica los conocimientos aprendidos durante este curso y sobre todo en la asignatura de Proyecto (M14 - Desarrollo de una aplicación web) ya que en ella hemos aprendido a gestionar un proyecto desde 0. En general el objetivo principal del proyecto ha sido alcanzado, aunque existen muchas mejoras posibles para la aplicación.

*Carlos*

Me parece que es un buen proyecto, aunque se podrían ampliar funcionalidades, pero hemos tenido que ajustarnos al tiempo. Como comentan mis compañeros gracias a la asignatura de M14 hemos podido poner en practica lo aprendido durante el curso.

Como punto a mejorar sería poder hacer un mejor diseño, análisis y planificación del proyecto, pero como comentaba anteriormente la gran falta de tiempo hacía que tuviésemos que adaptarnos a la entrega final.

*Gloria Taboada*

Mi valoración personal es muy positiva, teniendo en cuenta que este proyecto cuenta con un gran peso en la parte de diseño, cosa que me encanta desarrollar.

Creo que una cosa fundamental a la hora de crear una aplicación es hacerla atractiva a la vista, facilitando así su navegación por los diferentes apartados.

Desde el principio hemos coincidido todos los compañeros en las herramientas con las que trabajaríamos para crear el proyecto, que son las aprendidas durante el curso, y cada uno es más habilidoso en unas o en otras, así que el reparto de tareas ha sido fácil.

## 4. Bibliografía

### Páginas web:

- Laravel – Manual de Laravel  
<https://laravel.com/>
- Bootstrap – Manual de Bootstrap  
<https://getbootstrap.com/>
- YouTube – Guía instalación Laravel multilenguaje  
<https://www.youtube.com/watch?v=rUI4O1kspvk>
- YouTube – Guía instalación Laravel con autenticación (usuarios y administradores)  
Parte 1: <https://www.youtube.com/watch?v=iKRLrJXNN4M>  
Parte 2: <https://www.youtube.com/watch?v=lr2nAD9UDGg>  
Parte 3: <https://www.youtube.com/watch?v=P8T3MjZPDdI>

## 5. Anexos

### 5.1. Plan de Marketing

#### 5.1.1 Estudio de mercado

##### Localización

La sede principal de la empresa seria en Terrassa, pero no solo nos limitamos a un público de esta población.

##### Características y tendencias del mercado

Las aplicaciones web es un sector económico que mueve millones de euros. El mundo ha evolucionado mucho en estos últimos años y a nivel de aplicaciones web sigue evolucionando, por eso cualquier empresa de servicios desea tener su aplicación web para usuarios o clientes.

A la hora de tener que desarrollar una página web los hoteles son más precavidos. No quieren tener una página poco funcional y con un diseño poco atractivo, desean una página web donde el cliente se sienta cómodo y sea de fácil uso

El mercado de las aplicaciones web para reserva de hoteles está muy desarrollado, pero la expectativa es que siga creciendo debido a la aparición de nuevos lenguajes de programación y necesidades del cliente.

##### Análisis de la demanda

Con la página web del hotel se pretende que el hotel pueda seguir realizando sus tareas sin tener que preocuparse por atender a los clientes para hacer reservas. Nuestro cliente es muy exigente y sabe que quiere tanto el cómo sus clientes. No dejará que sus clientes reserven de cualquier forma.

La demanda busca la comodidad y buen servicio con un presupuesto ajustado. Por eso es importante que nuestra aplicación web tenga un buen diseño y una buena funcionalidad.

##### Análisis de la competencia

En Terrassa hay más de una veintena de empresas que se dedican al diseño y creación de páginas web. Con esta cantidad de empresas realizando las mismas funciones se han de realizar esfuerzos para mantener la posición en el mercado.

A parte de realizar un gran trabajo se han de realizar campañas publicitarias para diferenciar los productos que se ofrecen para poder competir con los precios de la competencia o la mejora del servicio a nuestros clientes

Viabilidad comercial (DAFO)

- Puntos fuertes:

- Formación académica sobre desarrollo de aplicaciones web que hemos recibido estos dos últimos años en el instituto Nicolau Copèrnic.
- Poca inversión económica
- Alta motivación del proyecto
- Somos personas jóvenes y entusiastas.

- Puntos débiles:

- No tener conocimientos en tener un negocio propio

- Oportunidades:

- La confianza de los clientes en las aplicaciones web en sus negocios

- Amenazas:

- La buena posición de las empresas ya instaladas
- La fuerte competencia del sector
- El gran poder financiero de las grandes empresas



### 5.1.2 Estrategia de marketing

#### El producto

La mayoría de los que buscan un hotel miran en Internet antes que en una agencia de viajes. 8 de cada 10 personas opinan que la red es el medio más fiable para conocer o ampliar información sobre una marca o producto. Nuestro producto está repartido en los siguientes puntos:

- La web se visualizará desde cualquier dispositivo móvil de forma que la búsqueda de información y la compra desde smartphones y Tablet sean experiencias especialmente cómodas y sencillas.
- Mejora de la visibilidad de la página haciendo que su contenido sea más legible y atractivo para los buscadores, posicionándose así en las primeras posiciones de búsqueda
- Gestionamos de forma gratuita la contratación del hosting y la disponibilidad y el registro del nombre de dominio de la web.
- Consultoría web de desarrollo a lo largo de todo el proceso y resoluciones técnicas durante el primer mes sin coste adicional alguno.

#### El precio

Todos nuestros servicios cuentan con precios independientes para todos los clientes. El precio medio por web profesional será de 249€.

#### La promoción

Los instrumentos a seguir respecto a los clientes será la del descuento en los precios. En la campaña de introducción se hará un descuento del 10% en todos sus productos durante los primeros dos meses de la inauguración, para incentivar al público a conocernos.

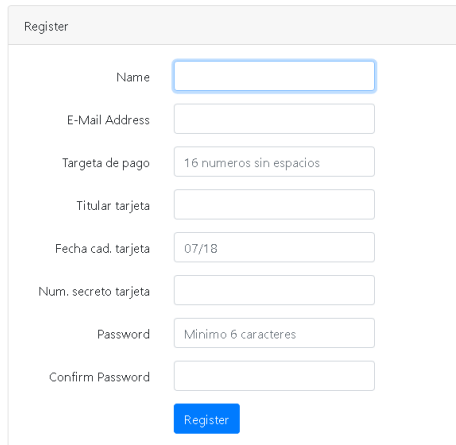
#### La distribución

Funcionará como una empresa mayorista. Su función será la de vender productos al minorista, para que este los ofrezca al consumidor final.

## 5.2. Manual de usuario

### -Registro de usuario estándar

1. Rellena los campos correctamente.
2. Clicka el botón de registro.



Register

Name

E-Mail Address

Targeta de pago

Titular tarjeta

Fecha cad. tarjeta

Num. secreto tarjeta

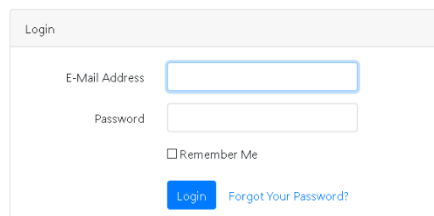
Password

Confirm Password

[Register](#)

### -Inicio de sesión de usuario

1. Introduce tus datos de usuario.
2. Clicka en botón de iniciar sesión.



Login

E-Mail Address

Password

☐ Remember Me

[Login](#) [Forgot Your Password?](#)

### -Mostrar Panel Usuario

1. Clicka en “Mi Perfil” para acceder a tus datos de usuario.
- 1.2. Clicka en “Mis Reservas” para ver tus reservas y poder cancelarlas.
- 1.3. Clicka en tu usuario y en el desplegable en “Cerrar sesión” para cerrar la sesión de tu usuario.



 **GRAN HOTEL MIRAMAR**

[Prueba](#)  [Español](#)

[Inicio](#)  
[Cerrar sesión](#)

**Tablero de usuario**

 **Mi Perfil**

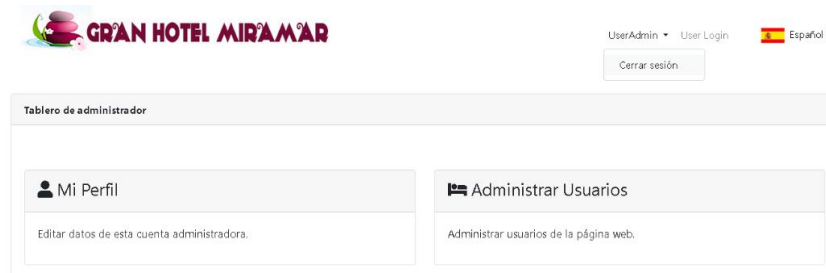
Editar datos personales y de cuenta de usuario.

 **Mis Reservas**

Realizar el seguimiento o cancelar una reserva

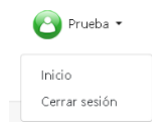
## -Mostrar Panel Administrador

1. Clica en “Mi Perfil” para acceder a tus datos de usuario administrador.
- 1.2. Clica en “Administrar Usuarios” para ver los usuarios de la aplicación y poder editar sus datos y darlos de baja.
- 1.3. Clica en tu usuario y en el desplegable en “Cerrar sesión” para cerrar la sesión de tu usuario administrador.



## -Cerrar sesión

1. Clica en tu usuario y en el desplegable en “Cerrar sesión” para cerrar la sesión de tu usuario estándar o administrador.



## -Editar Perfil

1. Clica en “Mi Perfil” para acceder a tus datos de usuario estándar o administrador.



2. Edita los datos que consideres.

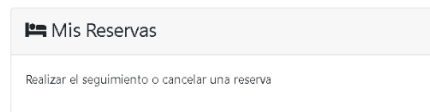
USUARIO ESTANDAR

USUARIO ADMINISTRADOR

3. Clica guardar para modificar tus datos de usuario estándar o administrador.

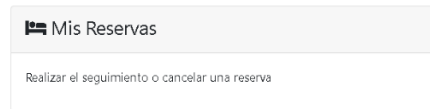
## -Mostrar reservas

1. Clicka en “Ver Reservas” para poder ver tus reservas.



## -Gestionar reservas

1. Clicka en “Ver Reservas” para poder ver tus reservas.

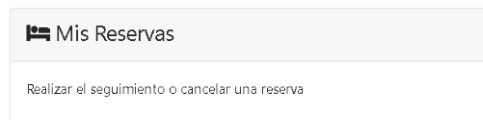


2. Puedes revisar tus reservas y cancelarlas.

Mis reservas				
Tipo reserva	Info reserva	Fecha entrada	Fecha salida   Hora	Opciones
Restaurante	Spring Space	01/06/2018	13:00	Cancelar reserva
Cerrar				

## -Cancelar reserva

1. Clicka en “Ver Reservas” para poder ver tus reservas.



2. Clicka en cancelar reserva.

Mis reservas				
Tipo reserva	Info reserva	Fecha entrada	Fecha salida   Hora	Opciones
Restaurante	Spring Space	01/06/2018	13:00	Cancelar reserva
Cerrar				

3. Confirma la cancelación de la reserva.

Cancelar reserva

¿Está seguro de que desea cancelar la reserva?

Cancelar reserva

Cerrar

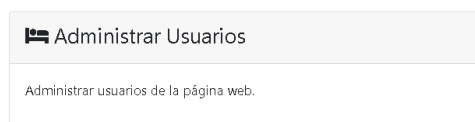
## -Mostrar usuarios (administrador)

1. Clica en “Administrar Usuarios” para ver los usuarios de la aplicación y poder editar sus datos y darlos de baja.

Administrar Usuarios						
Nombre	Correo electrónico	Número tarjeta	Titular tarjeta	Número secreto tarjeta	Fecha caducidad tarjeta	
no	k@k.com	1234567890123456				 Eliminar usuario
Prueba	prueba@prueba.com	1234567890123455	Oscar Robblaaaaaas	556	02/22	 Eliminar usuario
<div> <div>Guardar</div> <div>Cerrar</div> </div>						

## -Administrar usuarios (administrador)

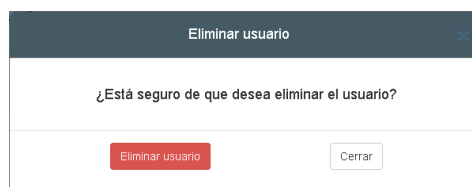
1. Clica en “Administrar Usuarios” para ver los usuarios de la aplicación.



2.1. Clica en “Eliminar usuario” para darlo de baja.

Administrar Usuarios						
Nombre	Correo electrónico	Número tarjeta	Titular tarjeta	Número secreto tarjeta	Fecha caducidad tarjeta	
no	k@k.com	1234567890123456				 Eliminar usuario
Prueba	prueba@prueba.com	1234567890123455	Oscar Robblaaaaaas	556	02/22	 Eliminar usuario
<div> <div>Guardar</div> <div>Cerrar</div> </div>						

2.1.1 Confirma la eliminación de usuario.

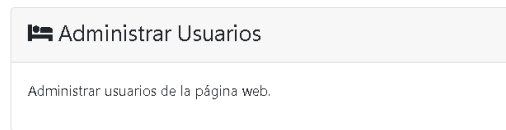


2.2. Edita los campos de los datos de usuario que desees modificar.

2.2.1. Clica en “Guardar para modificar los datos editados.

## -Eliminar usuario (administrador)

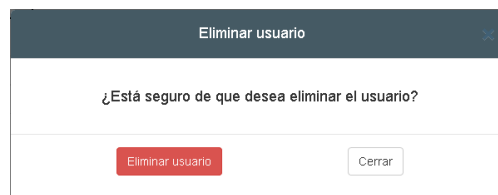
1. Clica en “Administrar Usuarios” para ver los usuarios de la aplicación.



2. Clica en “Eliminar usuario” para darlo de baja.

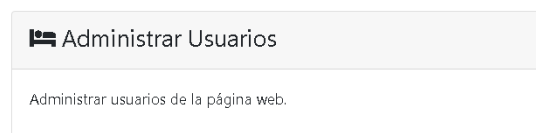
Administrar Usuarios						
Nombre	Correo electrónico	Número tarjeta	Titular tarjeta	Número secreto tarjeta	Fecha caducidad tarjeta	
no	k@k.com	1234567890123456				<button>Eliminar usuario</button>
Prueba	prueba@prueba.com	1234567890123455	Oscar Robblaaaaaas	556	02/22	<button>Eliminar usuario</button>
<div><button>Guardar</button><button>Cerrar</button></div>						

3. Confirma la eliminación de usuario.



## -Crear usuario (administrador)

1. Clica en “Administrar Usuarios” para ver los usuarios de la aplicación.



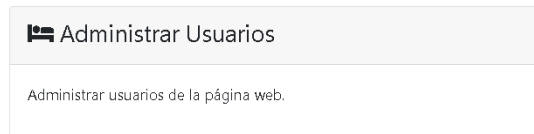
2. Clica en “Crear usuario” para darlo de alta.

3. Introduce sus datos.

4. Clica en “guardar” para crear el usuario.

### -Modificar usuario (administrador)

1. Clica en “Administrar Usuarios” para ver los usuarios de la aplicación.



2. Edita los campos de los datos de usuario que desees modificar.
3. Clica en “Guardar para modificar los datos editados.

### -Cambiar idioma

1. Clica en el selector de idiomas para ver los diferentes idiomas de la aplicación.



2. Selecciona el idioma que desees.

**-Reservar (habitación, restaurante, tratamiento, spa)**

1. Clica en el botón “Reservar” del servicio que desees reservar.



2. Introducimos los datos necesarios del formulario.

Reserva de Suite Presidencial

Habitación  
Suite Presidencial

Fecha Llegada  
Selecciona la fecha

Fecha Salida  
Selecciona la fecha

Huéspedes  
1

Numero tarjeta de pago  
16 numeros sin espacios

Titular tarjeta

Fecha cad. tarjeta  
07/18

Num. secreto tarjeta

Volver Reservar

3. Clica en “Reservar” para finalizar la reserva.



## 5.3. Manual de instalación

### Guía instalación Proyecto Laravel y GitHub

- Requisitos:

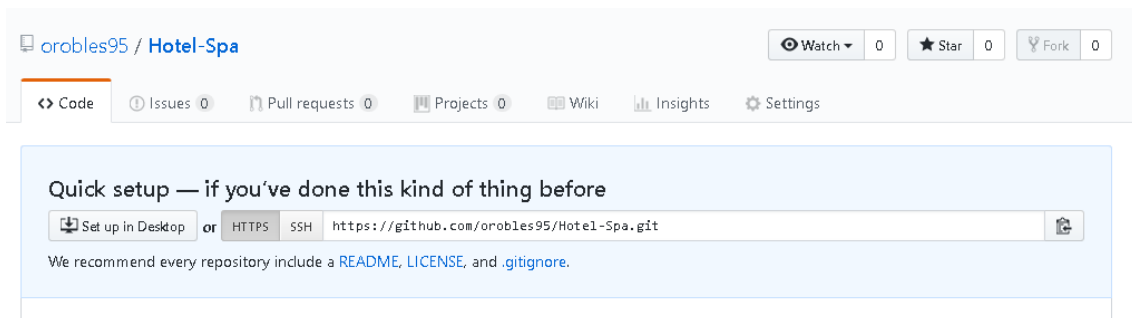
-Necesitaremos tener instalados los siguientes programas en nuestro equipo: Git, NetBeans Composer, Mycrypt y XAMPP.

- Pasos:

1. Abrimos símbolo de sistema o la consola de GitBash y creamos nuevo proyecto en “xampp/htdocs”:

```
oscar@MI-NOTEBOOK-OSCAR MINGW64 /c/xampp/htdocs
$ composer create-project laravel/laravel Hotel-Spa
Installing laravel/laravel (v5.6.12)
```

2. Creamos nuevo repositorio en GitHub (<https://github.com/orobles95/Hotel-Spa>) para trabajar con el proyecto.



3. Ahora iniciamos la consola de GitHub desde el proyecto para subirlo a nuestro repositorio GitHub. Empezamos ejecutando un “git init” preparar el directorio y crear las carpetas necesarias para compartir el proyecto.

Seguidamente ejecutamos un “git add .” para añadir todos los directorios y archivos contenidos en la carpeta del proyecto:

```
oscar@MI-NOTEBOOK-OSCAR MINGW64 /c/xampp/htdocs/Hotel-Spa (master)
$ git add .
```

A continuació, executamos el primer *commit* para poder realizar un *push* de todos los archivos:

```
oscar@MI-NOTEBOOK-OSCAR MINGW64 /c/xampp/htdocs/Hotel-Spa (master)
$ git commit -m "Primer commit"
[master (root-commit) 162f80c] Primer commit
```

Finalmente le indicamos la dirección de nuestro repositorio GitHub y ejecutamos un *push* de la raíz de nuestro proyecto:

```
oscar@MI-NOTEBOOK-OSCAR MINGW64 /c/xampp/htdocs/Hotel-Spa (master)
$ git remote add origin https://github.com/orobles95/Hotel-Spa.git

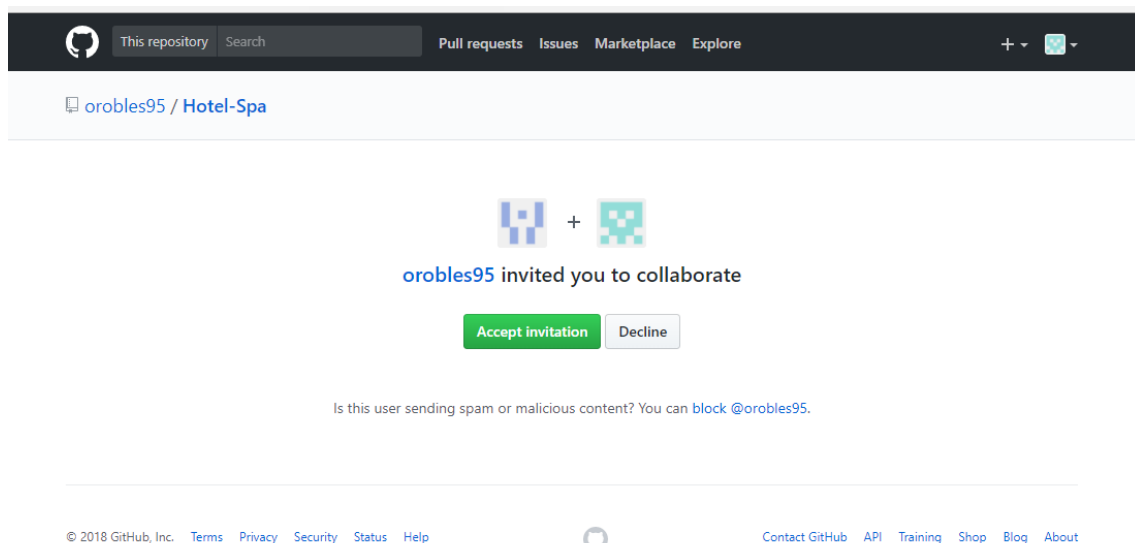
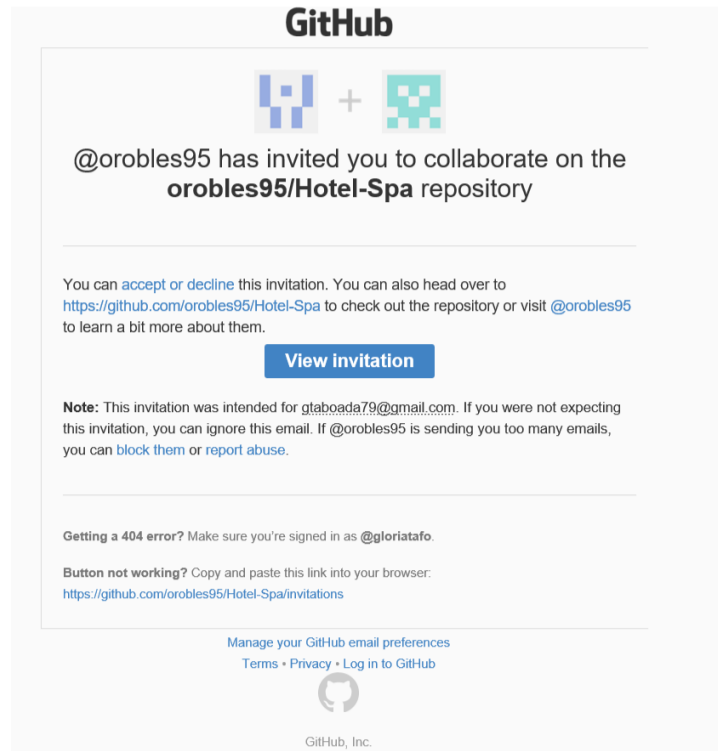
oscar@MI-NOTEBOOK-OSCAR MINGW64 /c/xampp/htdocs/Hotel-Spa (master)
$ git remote -v
origin https://github.com/orobles95/Hotel-Spa.git (fetch)
origin https://github.com/orobles95/Hotel-Spa.git (push)

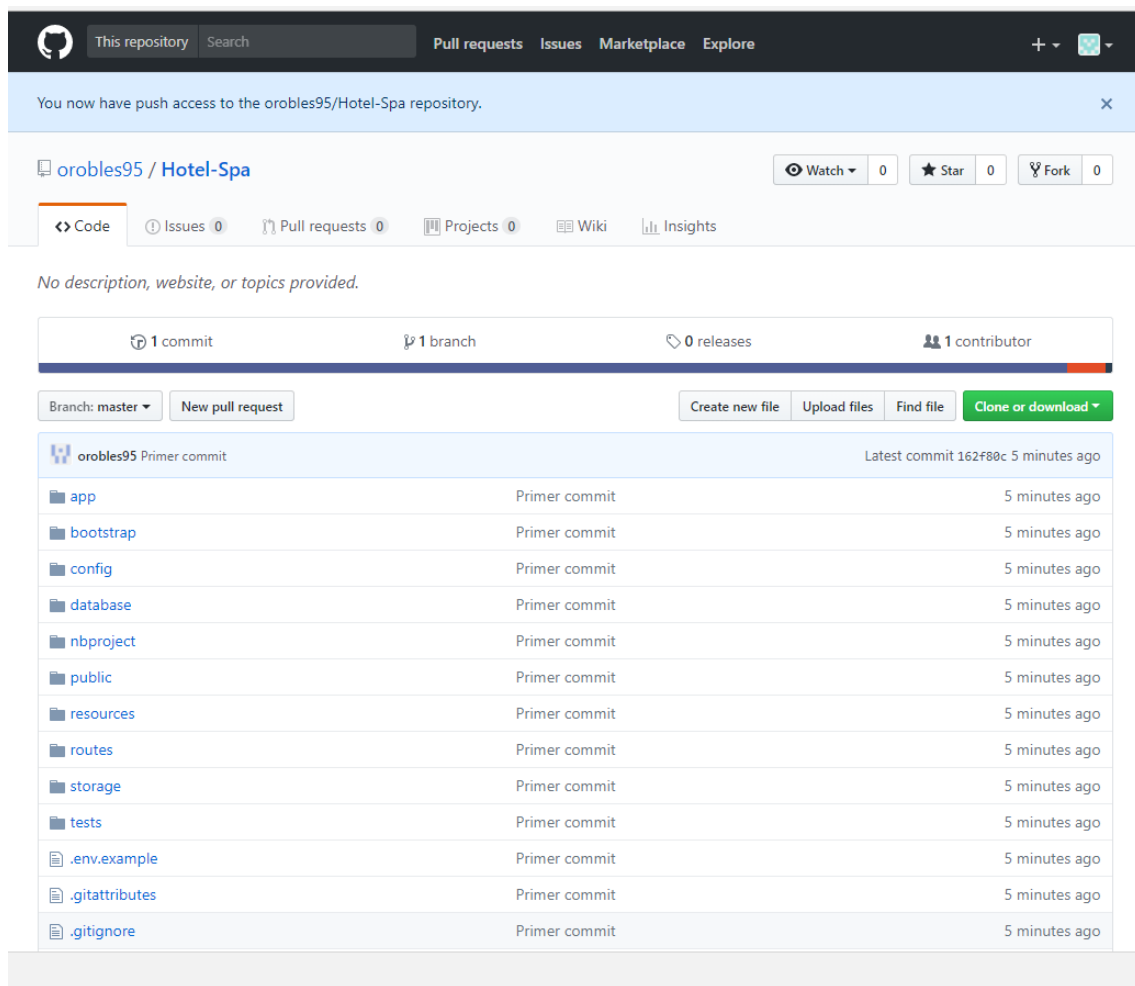
oscar@MI-NOTEBOOK-OSCAR MINGW64 /c/xampp/htdocs/Hotel-Spa (master)
$ git push origin master
Counting objects: 117, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (99/99), done.
Writing objects: 100% (117/117), 181.58 KiB | 3.08 MiB/s, done.
Total 117 (delta 9), reused 0 (delta 0)
remote: Resolving deltas: 100% (9/9), done.
To https://github.com/orobles95/Hotel-Spa.git
 * [new branch]      master -> master

oscar@MI-NOTEBOOK-OSCAR MINGW64 /c/xampp/htdocs/Hotel-Spa (master)
```

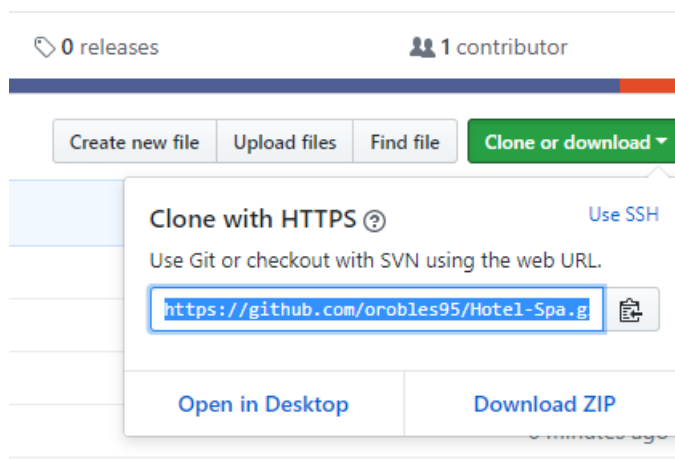
## 4. El compañero que no ha creado el proyecto debe seguir los siguientes pasos:

### 4.1. Unirse al repositorio del proyecto en GitHub (en este caso con invitación del creador del repositorio).

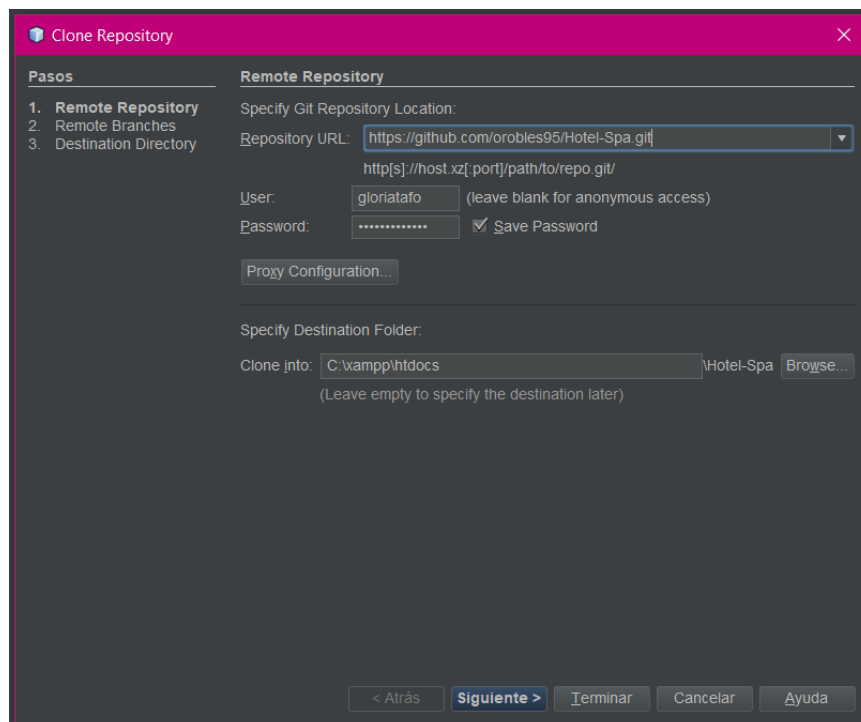
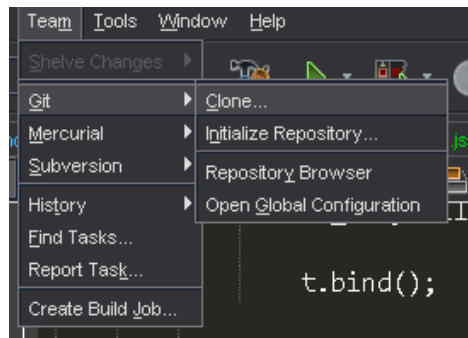




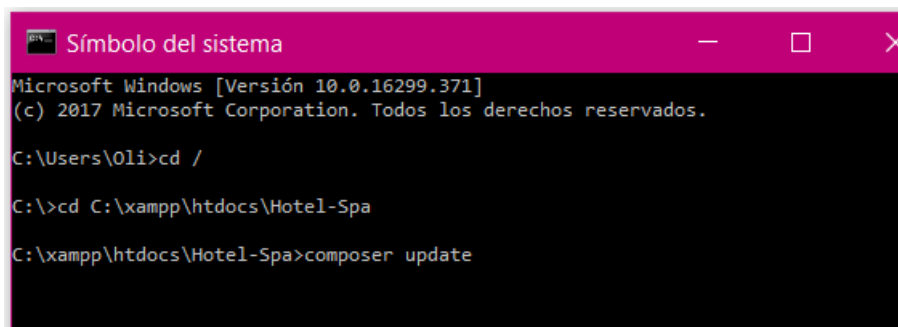
4.2. Después tendremos que clonar el proyecto. Existen varias formas de lograr esto, pero nosotros utilizaremos la dirección del repositorio:



4.3. A continuació, obrim NetBeans y accedemos a la siguiente opción:



4.4. Ahora desde el terminal deberemos acceder a la carpeta donde clonemos nuestro proyecto para actualizar Composer:



4.5. Lo siguiente que debemos hacer es duplicar el archivo “env.example” del proyecto y cambiarle el nombre como “env.” Y configurar los siguientes datos con nuestra configuración de nuestro servidor, nuestra base de datos y usuario y contraseña para acceder a ella.

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=homestead
DB_USERNAME=root
DB_PASSWORD=
```

4.6. Por último, deberemos acceder de nuevo al terminal a nuestro proyecto y crear una nueva clave para la aplicación:

```
C:\xampp\htdocs\Hotel-Spa>php artisan key:generate
Application key [base64:tfD87WP+5g27NaYcsbWi78UzqVOjR9cq/9lW0VgC9o8=] set successfully.
C:\xampp\htdocs\Hotel-Spa>
```

4.7. Ahora ya podremos acceder a nuestro proyecto (por ejemplo, la vista por defecto):

# Laravel

[DOCUMENTATION](#)

[LARACASTS](#)

[NEWS](#)

[FORGE](#)

[GITHUB](#)

## Guía instalación autenticación de usuarios

1. En terminal creamos los archivos necesarios para la autenticación en u proyecto Laravel:

```
CA: Símbolo del sistema
Microsoft Windows [Versión 10.0.17134.81]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Vayava>cd ..\..\xampp\htdocs\Hotel-Spa

C:\xampp\htdocs\Hotel-Spa>php artisan make:auth
Authentication scaffolding generated successfully.

C:\xampp\htdocs\Hotel-Spa>
```

2. Añadimos la table admin a las migraciones y creamos las tablas por defecto de usuarios, reset de contraseña y la tabla admin creada:

```
CA: Símbolo del sistema
Microsoft Windows [Versión 10.0.17134.81]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Vayava>cd ..\..\xampp\htdocs\Hotel-Spa

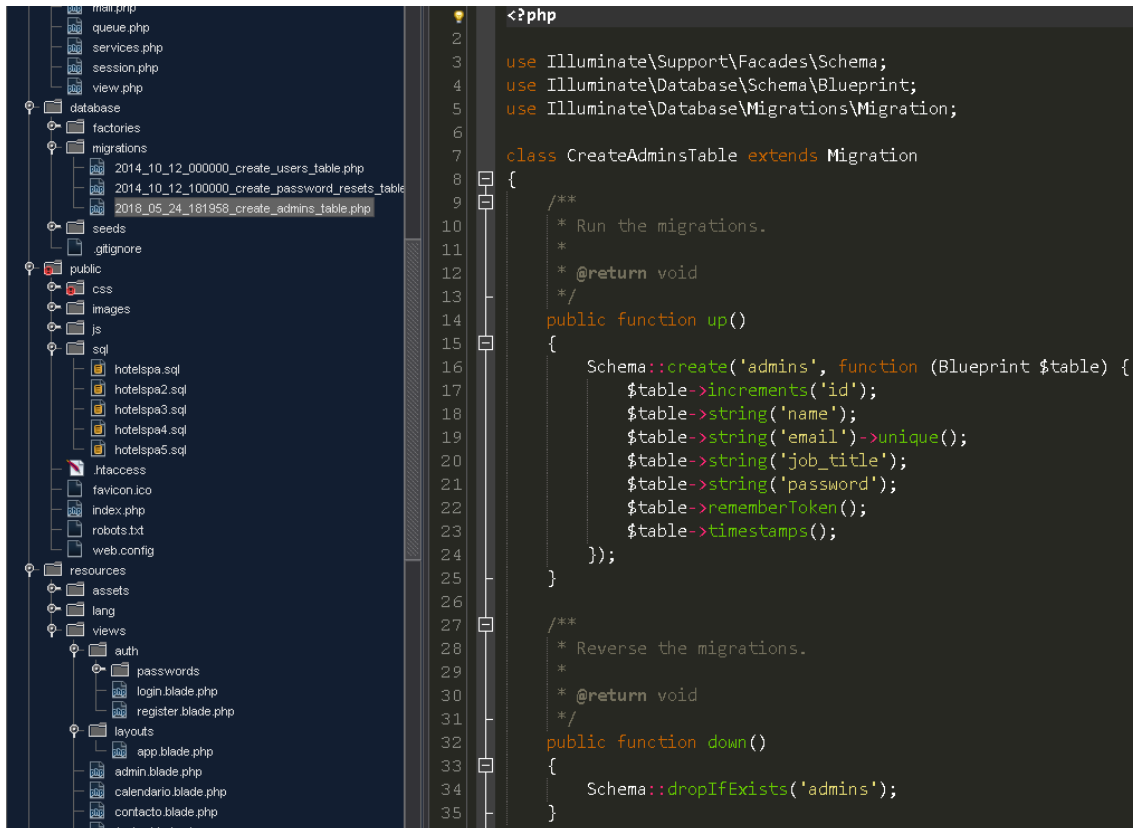
C:\xampp\htdocs\Hotel-Spa>php artisan make:auth
Authentication scaffolding generated successfully.

C:\xampp\htdocs\Hotel-Spa>php artisan make:migration create_admins_table --create=admins
Created Migration: 2018_05_24_181958_create_admins_table

C:\xampp\htdocs\Hotel-Spa>php artisan migrate
Migration table created successfully.
Migrating: 2014_10_12_000000_create_users_table
Migrated: 2014_10_12_000000_create_users_table
Migrating: 2014_10_12_100000_create_password_resets_table
Migrated: 2014_10_12_100000_create_password_resets_table
Migrating: 2018_05_24_181958_create_admins_table
Migrated: 2018_05_24_181958_create_admins_table

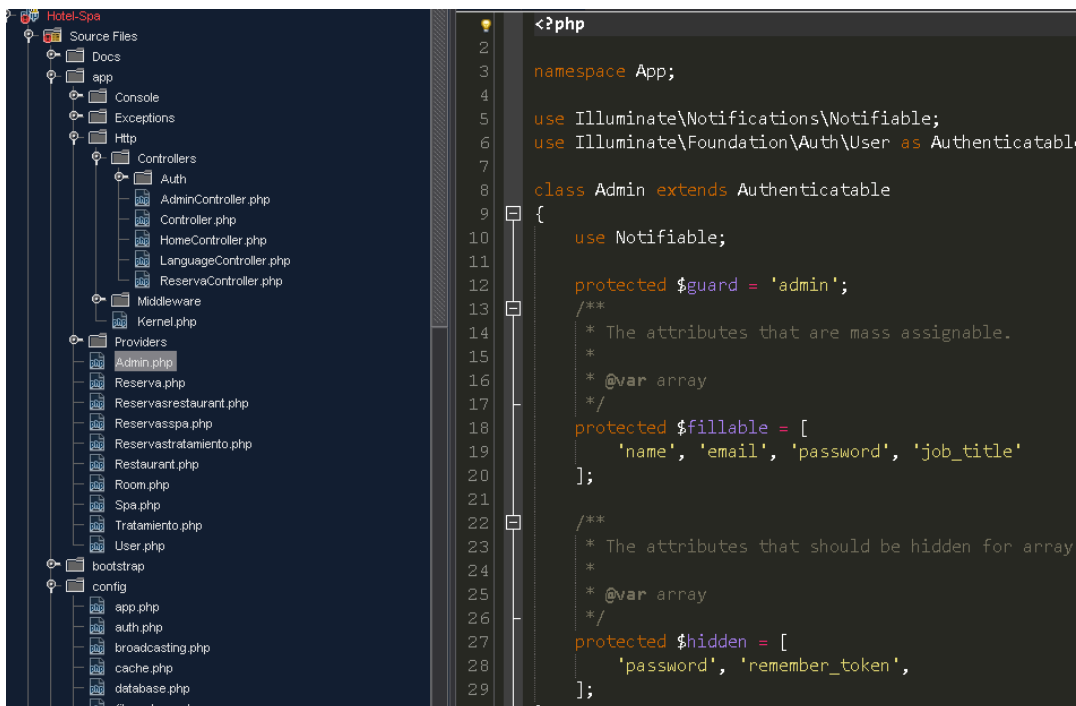
C:\xampp\htdocs\Hotel-Spa>_
```

3. Editamos la migración de “admin” con los mismos datos que un usuario normal además de un campo “job\_title” destinado a la función de ese administrador:



```
<?php
2
3 use Illuminate\Support\Facades\Schema;
4 use Illuminate\Database\Schema\Blueprint;
5 use Illuminate\Database\Migrations\Migration;
6
7 class CreateAdminsTable extends Migration
8 {
9     /**
10      * Run the migrations.
11      *
12      * @return void
13      */
14     public function up()
15     {
16         Schema::create('admins', function (Blueprint $table) {
17             $table->increments('id');
18             $table->string('name');
19             $table->string('email')->unique();
20             $table->string('job_title');
21             $table->string('password');
22             $table->rememberToken();
23             $table->timestamps();
24         });
25     }
26
27     /**
28      * Reverse the migrations.
29      *
30      * @return void
31      */
32     public function down()
33     {
34         Schema::dropIfExists('admins');
35     }
36 }
```

4. Accedemos al modelo de Admin para seleccionar la tabla “admin” al usar el modelo Admin y la editamos con los siguientes datos:



```
<?php
2
3 namespace App;
4
5 use Illuminate\Notifications\Notifiable;
6 use Illuminate\Foundation\Auth\User as Authenticatable;
7
8 class Admin extends Authenticatable
9 {
10     use Notifiable;
11
12     protected $guard = 'admin';
13
14     /**
15      * The attributes that are mass assignable.
16      *
17      * @var array
18      */
19     protected $fillable = [
20         'name', 'email', 'password', 'job_title'
21     ];
22
23     /**
24      * The attributes that should be hidden for array
25      *
26      * @var array
27      */
28     protected $hidden = [
29         'password', 'remember_token',
30     ];
31 }
```



5. A continuació, accedem a “config/auth.php” y editamos las siguientes secciones añadiendo la configuración necesarios para nuestro administrador:



```

35
36
37
38 'guards' => [
39     'web' => [
40         'driver' => 'session',
41         'provider' => 'users',
42     ],
43
44     'api' => [
45         'driver' => 'token',
46         'provider' => 'users',
47     ],
48
49     'admin' => [
50         'driver' => 'session',
51         'provider' => 'admins',
52     ],
53
54     'admin-api' => [
55         'driver' => 'token',
56         'provider' => 'admins',
57     ],
58 ],

```

```

'providers' => [
    'users' => [
        'driver' => 'eloquent',
        'model' => App\User::class,
    ],
    'admins' => [
        'driver' => 'eloquent',
        'model' => App\Admin::class,
    ],
],

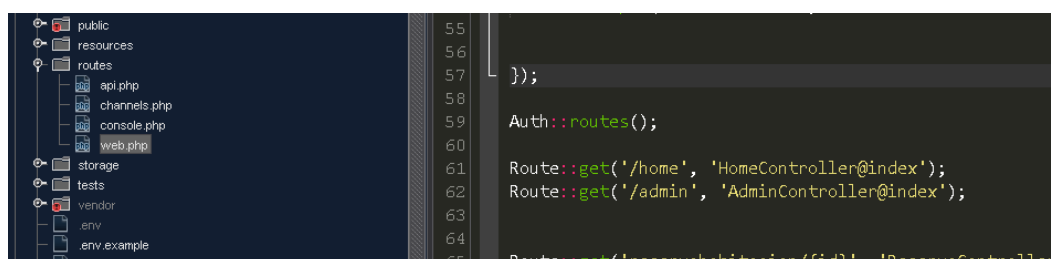
```

```

'passwords' => [
    'users' => [
        'provider' => 'users',
        'table' => 'password_resets',
        'expire' => 60,
    ],
    'admins' => [
        'provider' => 'admins',
        'table' => 'password_resets',
        'expire' => 15,
    ],
],

```

6. Ahora accedemos al archivo de rutas de Laravel (“routes/web.php”) para crear la protección al acceder a las vistas de usuario, admin y el resto de rutas de inicio de sesión y registro relacionadas con la autenticación de usuarios:

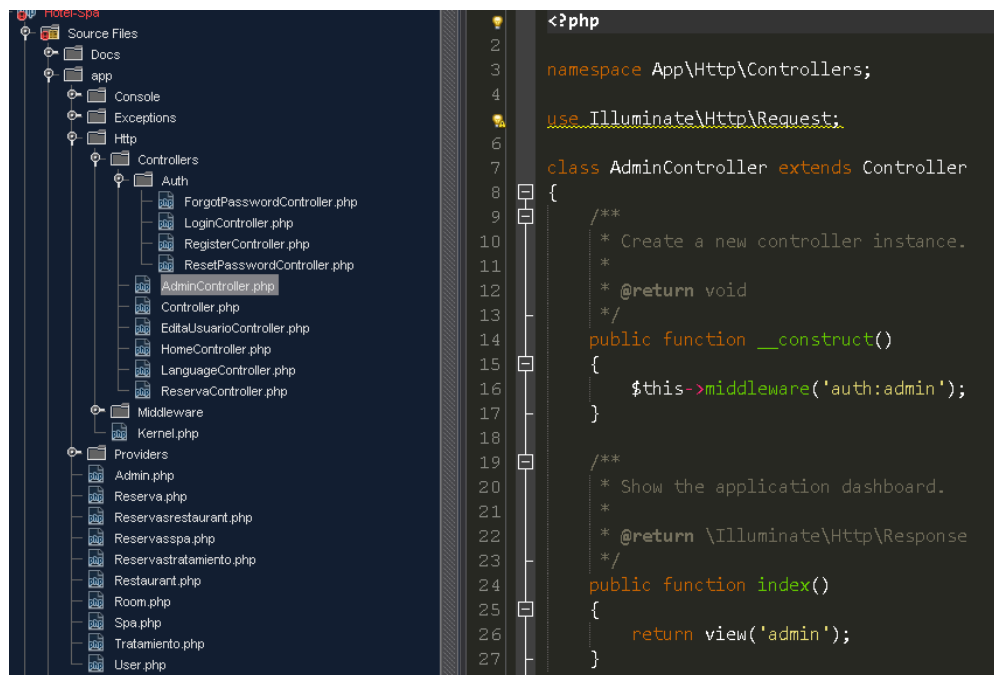


```

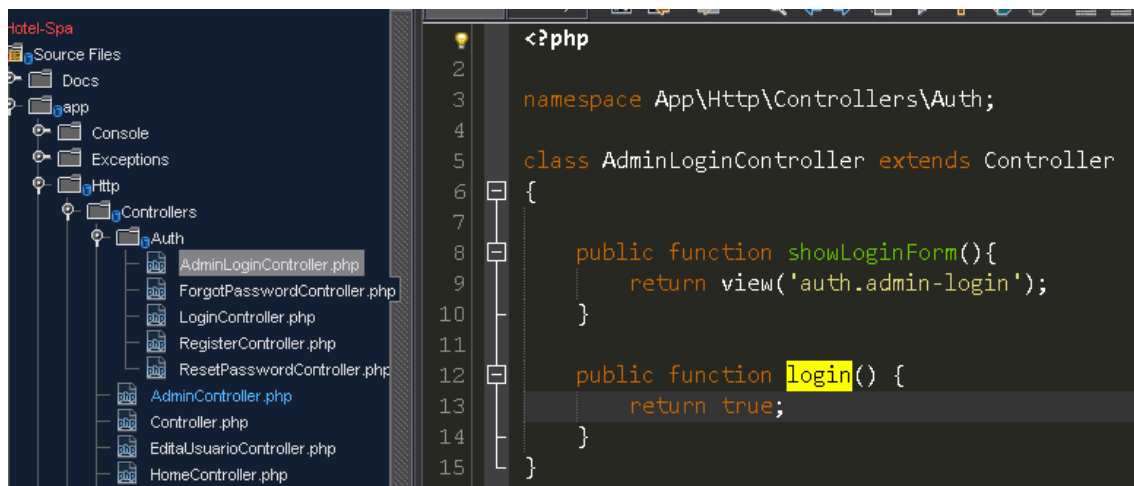
55
56
57
58
59 Auth::routes();
60
61 Route::get('/home', 'HomeController@index');
62 Route::get('/admin', 'AdminController@index');
63
64 Route::get('/reservahabitacion/{id}', 'ReservaController
65

```

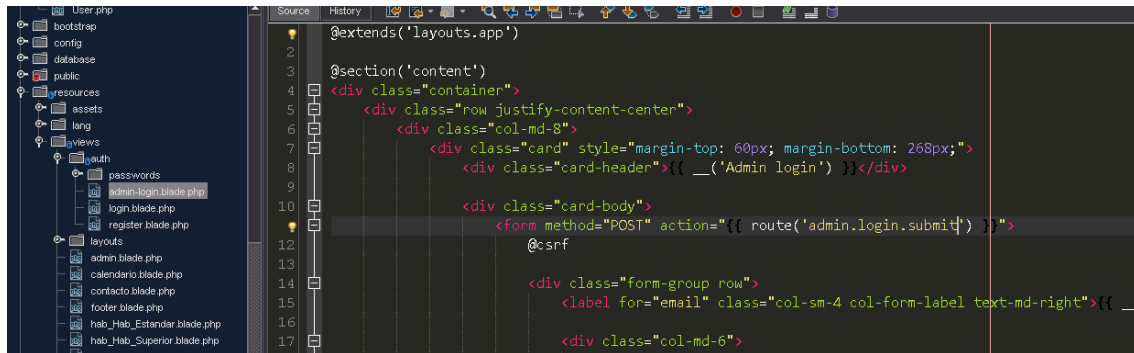
7. Creamos el controlador que usara la vista de administrador agregando en esta la especificación de que solo pueda acceder un usuario de la tabla “admin”:



8. A continuación, creamos el controlador de login del administrador en “app/Http/Controllers/AdminLoginController.php”:



9. A continuació creamos la vista de login de administrador en “resources/views/auth/admin-login.blade.php” y editamos la ruta al enviar el formulario como se ve en la imagen:



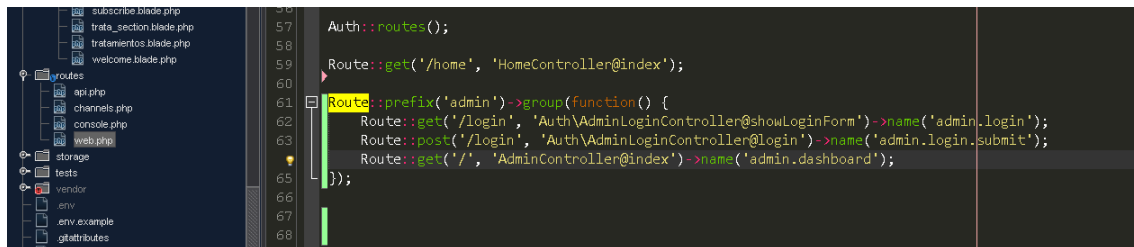
```

@extends('layouts.app')

@section('content')
    <div class="container">
        <div class="row justify-content-center">
            <div class="col-md-8">
                <div class="card" style="margin-top: 60px; margin-bottom: 268px;">
                    <div class="card-header">{{ __('Admin login') }}</div>
                    <div class="card-body">
                        <form method="POST" action="{{ route('admin.login.submit') }}">
                            @csrf
                            <div class="form-group row">
                                <label for="email" class="col-sm-4 col-form-label text-md-right">{{ __('Email') }}</label>
                                <div class="col-md-6">
                                    <input type="text" class="form-control" value="{{ old('email') }}">
                                </div>
                            </div>
                            <div class="form-group row">
                                <label for="password" class="col-sm-4 col-form-label text-md-right">{{ __('Password') }}</label>
                                <div class="col-md-6">
                                    <input type="password" class="form-control" value="{{ old('password') }}">
                                </div>
                            </div>
                            <div class="form-group row">
                                <div class="col-sm-4">
                                    <input type="checkbox" class="checkbox" value="1"> {{ __('Remember me') }}
                                </div>
                                <div class="col-md-6">
                                    <input type="button" value="{{ __('Login') }}" class="btn btn-primary">
                                </div>
                            </div>
                        </form>
                    </div>
                </div>
            </div>
        </div>
    </div>

```

10. Ahora editaremos el fichero de rutas con las siguientes:



```

Auth::routes();

Route::get('/home', 'HomeController@index');

Route::prefix('admin')->group(function() {
    Route::get('/login', 'Auth\AdminLoginController@showLoginForm')->name('admin.login');
    Route::post('/login', 'Auth\AdminLoginController@login')->name('admin.login.submit');
    Route::get('/', 'AdminController@index')->name('admin.dashboard');
});

```

11. Terminando editamos el controlador de login de administrador “app/Http/controllers/AdminLoginController.php”:



```

<?php

namespace App\Http\Controllers\Auth;

use Illuminate\Http\Request;
use App\Http\Controllers\Controller;
use Auth;

class AdminLoginController extends Controller {

    public function __construct() {
        $this->middleware('guest:admin');
    }

    public function showLoginForm() {
        return view('auth.admin-login');
    }

    public function login(Request $request) {


        $this->validate($request, [
            'email' => 'required|email',
            'password' => 'required|min:6',
        ]);

        if (Auth::guard('admin')->attempt(['email' => $request->email, 'password' => $request->password], $request->remember)) {
            return redirect()->intended(route('admin.dashboard'));
        }

        return redirect()->back()->withInput($request->only('email','remember'));
    }
}

```

12. Por último, solo nos queda crear un administrador en nuestra tabla Admin desde el terminal para probar que el código funciona correctamente:

 Símbolo del sistema - php artisan tinker

```
C:\xampp\htdocs\Hotel-Spa>php artisan tinker
Psy Shell v0.9.4 (PHP 7.2.1 - cli) by Justin Hileman
>>> $admin = new App\Admin
=> App\Admin {#2342}
>>> $admin->name = "Oscar"
=> "Oscar"
>>> $admin->email = "admin@hotelspa.com"
=> "admin@hotelspa.com"
>>> $admin->password = Hash::make('adminhotelspa')
=> "$2y$10$DZ.nIS.98xOJn2yG6PO100vwu.06mQ2JjSH05KxKIuH6a37ZYTkpw"
>>> $admin->job_title = "Administrador_BBDD"
=> "Administrador_BBDD"
>>> $admin->save()
Illuminate/Database/QueryException with message 'SQLSTATE[HY000]: General error
value (SQL: insert into `admins` (`name`, `email`, `password`, `job_title`,
admin@hotelspa.com, $2y$10$DZ.nIS.98xOJn2yG6PO100vwu.06mQ2JjSH05KxKIuH6a37ZY
, 2018-05-30 17:34:46))'
```

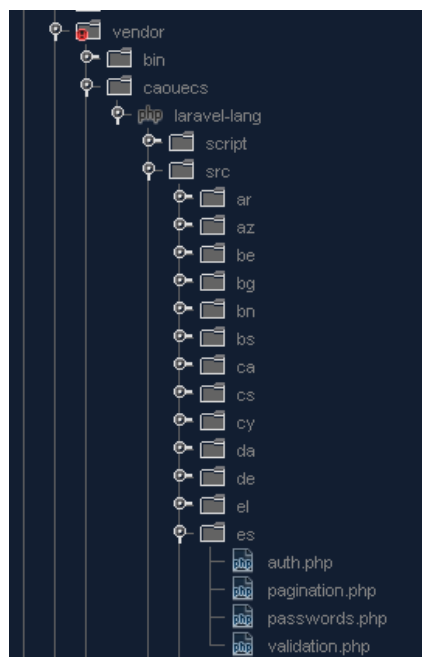
13. Finalmente, modificamos el acceso a el dashboard de administrador para que quede accesible si ya hemos iniciado sesión cuando accedamos a la vista de “login user” ubicada en “resources/views/layouts/app.php”:

## Guía instalación proyecto multilingüe

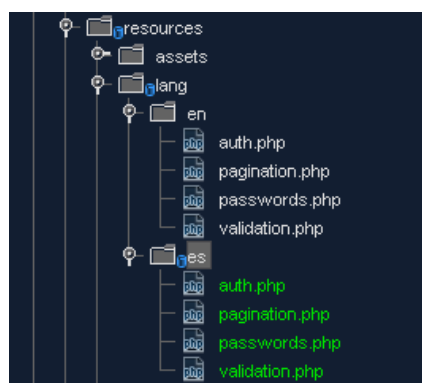
1. El primer paso es acceder a la carpeta raíz de nuestro proyecto y ejecutar el siguiente comando:

```
C:\WINDOWS\system32\cmd.exe - composer require caouecs/laravel-lang:~3.0  
  
C:\xampp\htdocs\Hotel-Spa>composer require caouecs/laravel-lang:~3.0  
./composer.json has been updated  
Loading composer repositories with package information  
Updating dependencies (including require-dev)  
Package operations: 1 install, 18 updates, 0 removals  
- Updating symfony/css-selector (v4.0.8 => v4.0.9): Loading from cache
```

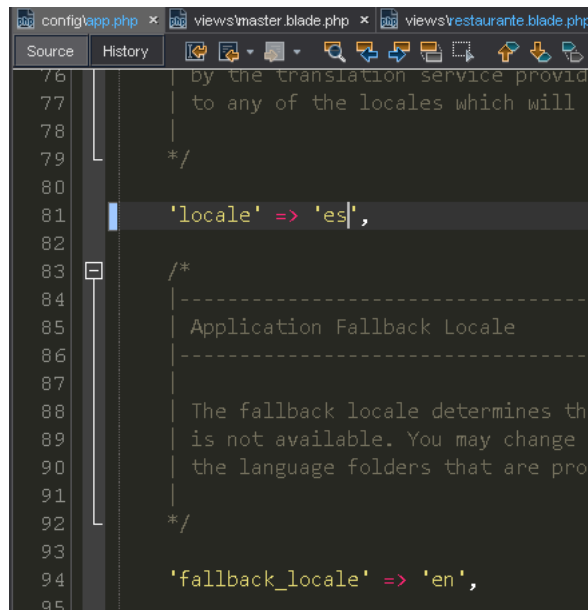
Esto nos creará las siguientes carpetas de varios idiomas en “vendor/caouecs/src” con los archivos necesarios predeterminados de autenticación y más configuraciones:



2. Ahora seleccionamos las que queramos implementar en nuestro proyecto y deberemos copiarlas en “resources/lang”:



3. Ahora podemos configurar nuestro idioma por defecto en nuestro proyecto en “config/app.php” cambiando el valor de ‘locale’ y seguidamente también podemos configurar el idioma secundario en caso de que no encuentre nuestra librería de idioma principal configurando el valor de ‘fallback\_locale’:

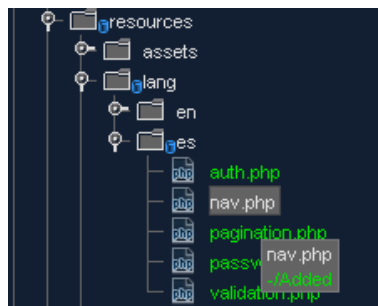


```

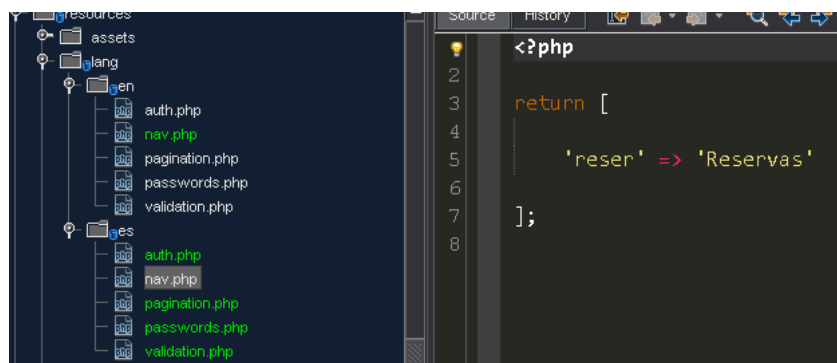
76 | by the translation service provide
77 | to any of the locales which will b
78 |
79 | */
80 |
81 | 'locale' => 'es',
82 |
83 | /*
84 | -----
85 | Application Fallback Locale
86 | -----
87 |
88 | The fallback locale determines the
89 | is not available. You may change t
90 | the language folders that are prov
91 | */
92 |
93 |
94 | 'fallback_locale' => 'en',
95 |

```

4. Ahora creamos el archivo “nav.php” que utilizaremos en nuestra primera vista traducida tanto en la carpeta ‘en’ (vista ingles) como en la ‘es’ (vista español):



5. Añadimos en estos archivos la siguiente estructura json con sus referencias “name=value”:





6. Añadimos en la vista la función php que utiliza nuestro archivo ‘nav’ y selecciona el valor de ‘reser’:

```
<a class="nav-link" href="#">
    <span>{{ trans('nav.reser') }}</span>
    php artisan make:controller LanguageController
Controller created successfully.

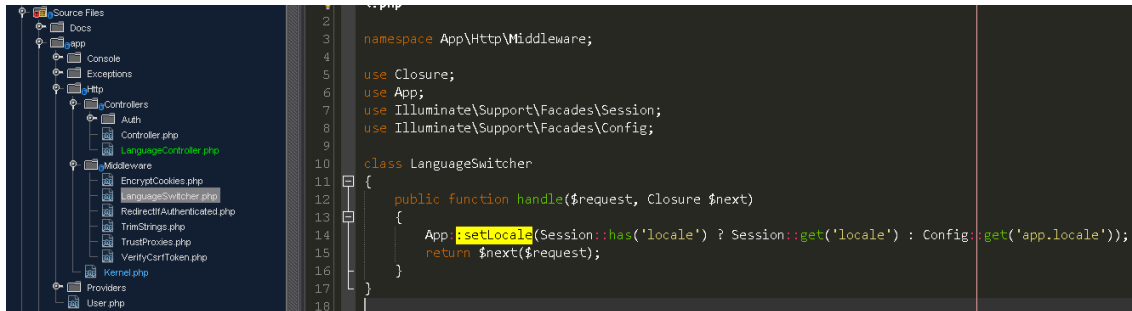
C:\xampp\htdocs\Hotel-Spa>php artisan make:middleware LanguageSwitcher
Middleware created successfully.

C:\xampp\htdocs\Hotel-Spa>
```

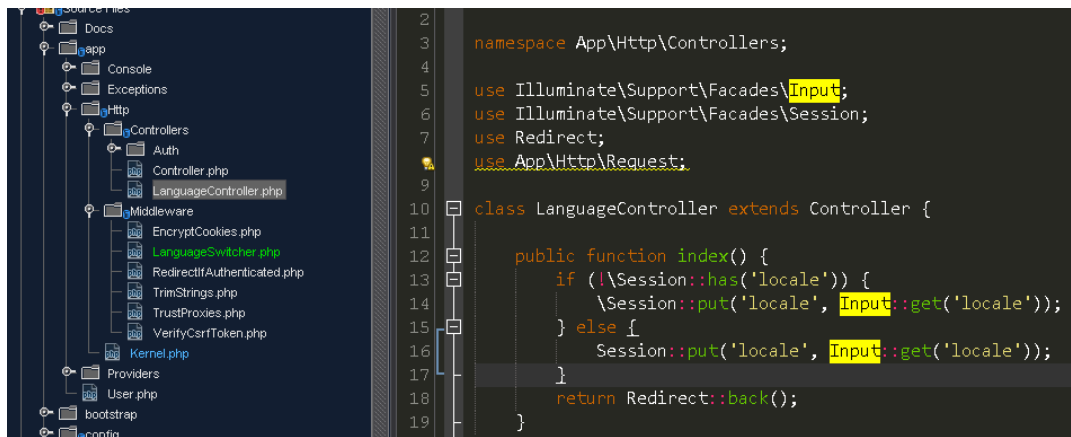
8. Ahora accedemos al archivo “app/Http/Kernel.php” y le añadimos las siguientes líneas:

9. Seguidamente configuramos el middleware y el controlador cambiador de idiomas creado anteriormente:

Middleware LanguageSwitcher:



Controlador LanguageController:



10. Por último, accedemos a la vista donde ubicaremos el seleccionador de idioma con un “select” y este será el código que utilizaremos:

