



# HOTEL RURAL SIERRA DE GATA

Proyecto Fin de Ciclo Grado Superior ASIR

Pablo Benítez Lajas  
Ciclo: 2º ASIR  
Curso: 2020-21  
Departamento: Informática  
Tutor: Jesús Antón Cuevas  
Centro: CIFP Ponferrada

# INDICE DE CONTENIDOS

<b>INDICE DE CONTENIDOS.....</b>	<b>1</b>
<b>1. INTRODUCCIÓN .....</b>	<b>2</b>
1.1 NOMBRE Y PRESENTACIÓN DEL PROYECTO .....	2
1.2 OBJETIVO.....	2
1.3 JUSTIFICACIÓN .....	2
1.4 EVALUACIÓN DE COSTES.....	3
<b>2. DISEÑO .....</b>	<b>4</b>
2.1 HERRAMIENTAS UTILIZADAS .....	4
2.2 LENGUAJES UTILIZADOS .....	5
2.3 DISEÑO ARQUITECTÓNICO .....	8
<b>3. BASE DE DATOS.....</b>	<b>10</b>
3.1 DISEÑO DE BASE DE DATOS.....	10
3.2 DEFINICIÓN DE TABLAS .....	11
3.3 RELACIONES .....	14
3.4 CONEXIÓN CON LA BBDD .....	15
<b>4. ARQUITECTURA DE RED .....</b>	<b>19</b>
4.1 ESQUEMA Y ESTRUCTURA DE LA RED.....	19
4.2 CONEXIÓN ENTRE OFICINAS/REDES .....	21
<b>5. DISEÑO WEB .....</b>	<b>25</b>
5.1 SECCIONES WEB.....	25
5.2 REGISTRO.....	31
5.3 INICIO DE SESIÓN .....	33
5.4 ACCESO USUARIOS.....	35
5.5 CUENTA DE EMPLEADO .....	39
<b>6. SERVIDOR WEB EN RASPBERRY .....</b>	<b>43</b>
6.1 ESPECIFICACIONES.....	43
6.2 INSTALACIÓN DEL HARDWARE .....	44
6.3 PASOS EN LA INSTALACIÓN .....	45
6.4 ACCESO DESDE CLIENTE SSH.....	47
6.5 INSTALACIÓN SERVIDOR WEB.....	47
6.6 ACCESO POR FTP AL SERVIDOR .....	51
6.7 DNS DINÁMICO CON NO-IP .....	55
<b>7. SEGURIDAD .....</b>	<b>57</b>
7.1 FORMULARIOS .....	57
7.2 COPIAS DE SEGURIDAD .....	57
<b>8. RECURSOS WEB .....</b>	<b>60</b>
8.1 YOUTUBE .....	60
8.2 GITHUB.....	60
<b>9. CONCLUSIONES .....</b>	<b>62</b>
<b>10. BIBLIOGRAFIA .....</b>	<b>63</b>

# 1. INTRODUCCIÓN

## 1.1 NOMBRE Y PRESENTACIÓN DEL PROYECTO

HOTEL RURAL SIERRA DE GATA, es un Hotel situado cerca de la Sierra de Gata (provincia de Cáceres).

Una empresa que lleva muchos años operando en Extremadura, pero que, debido a la falta de clientela durante el último año, decidió actualizar su modelo de negocio al siglo XXI, es decir, crear un sitio web donde se puedan satisfacer las necesidades de los visitantes, así como, atraer a más clientes a través de la publicidad de la misma.

## 1.2 OBJETIVO

La finalidad principal será construir un sitio web rápido, económico, dinámico y moderno, donde cualquier usuario que quiera contratar servicios con la empresa pueda registrarse en la web mediante un formulario simple y realizar/consultar sus reservas.

Así como, el diseño de la red donde se ubicarán los trabajadores del hotel gestionando diferentes aspectos relativos a la empresa, como, por ejemplo, marketing, ventas, cobros... etc.

Para todo ello, hemos decidido adquirir un hardware especializado, hemos presupuestado la compra de una Raspberry pi 3+ que usaremos como Servidor Web y como ubicación del sitio web.

## 1.3 JUSTIFICACIÓN

Debido a la situación actual y los problemas que encontramos en la pandemia, me ha parecido adecuado enfocar el proyecto en un tipo de negocio que busca principalmente que sus clientes desconecten de la rutina del día a día, disfruten de la naturaleza y conozcan y degusten la gastronomía local de la zona.

Además de utilizar todos los conocimientos adquiridos en estos dos cursos de ASIR para lograr los objetivos de la empresa, tendremos que investigar sobre tecnologías que implementaremos en la página web, base de datos o en el hardware que actuará como servidor web (Raspberry pi 3 +).

## 1.4 EVALUACIÓN DE COSTES

El hardware necesario para la empresa se ha adquirido en Amazon:

- [Raspberry Pi 3 Modelo B+](#). Con un procesador de cuatro núcleos a 1,4 GHz, memoria RAM DDR2 de 1 GB, 4 puertos USB 2.0, interfaz de tarjeta gráfica integrada... etc.

Hemos presupuestado 4 Raspberry (48€/ud):

- Dos actuarán como Servidor DHCP, uno en cada oficina.
- 1 servidor Web.
- 1 servidor BBDD.

- [Accesorios Raspberry](#): Incluye caja protectora, fuente de alimentación y disipadores térmicos. (x4).
- [Switch 8 puertos](#): Necesitaremos dos Switch para conectar los ordenadores a la red LAN de las dos oficinas. (x2).
- [Switch Ethernet con 5 Puertos](#): Un Switch sencillo para conectar los servidores, ya que, con el Switch anterior, ya tenemos todas las conexiones establecidas (7 para dispositivos y 1 para el router).
- [Tarjeta SD 32 GB](#): Suficiente para instalar el sistema operativo, servidor web, MySQL, alojar en ella la página web junto a sus componentes (archivos CSS, JS, imágenes, etc.) (x4)

Nuestro presupuesto para este apartado es el siguiente:

HARDWARE	PRECIO UNITARIO	UD	PRECIO TOTAL
<i>Raspberry Pi 3 Modelo B+</i>	48 €	4	192,00 €
<i>Accesorios Raspberry</i>	13 €	4	52,00 €
<i>Tarjeta SD 32 GB</i>	10 €	4	40,00 €
<i>Switch 8 Puertos</i>	20,99 €	2	41,98 €
<i>Switch 5 Puertos</i>	7,50 €	1	7,50 €
			333,48 €

Otros gastos derivados de la implementación de redes/web:

Otros Gastos	UD	PRECIO TOTAL
<i>Página Web</i>	1	700 €
<i>Diseño BBDD y RED</i>	1	200 €
<i>Puesta a punto de dispositivos</i>	1	100 €
		1.000,00 €

Gastos totales de la empresa:

PRESUPUESTADO	1.333,48 €
IVA APLICABLE (21%)	280,03 €
IMPORTE NETO TOTAL	1.613,51 €

Sumados todos los costes, incluido el IVA correspondiente, el precio final sería de **1613,51€**.

Además, hemos ofertado un servicio de mantenimiento y supervisión de la web por 20€ al mes.

## 2. DISEÑO

### 2.1 HERRAMIENTAS UTILIZADAS

- **XAMP:** Es una distribución de Apache, que incluye varios softwares libres como:
  - Servidor WEB Apache:** Es el servidor web más usado globalmente para la entrega de contenidos web. Es necesario instalarlo en el dispositivo que hará de servidor web.
  - **PHP.MYADMIN:** Es una herramienta de software gratuita escrita en PHP, destinada a manejar la administración de MySQL a través de la WEB. Con esta herramienta hemos creado la base de datos y sus relaciones.
- **NETBEANS:** Es un IDE (Entorno de desarrollo integrado) orientado al desarrollo de aplicaciones Java (normalmente), aunque en nuestro caso, lo hemos utilizado para realizar nuestro código PHP y JavaScript aplicado a la página.
- **CISCO PACKET-TRACER:** Es una herramienta desarrollada por Cisco, que simula el comportamiento de una red incluyendo todos sus dispositivos. Utilizada para simular la red a implantar en la empresa.
- **DIA:** Es un programa utilizado para dibujar diagramas de estructura de datos, con esta herramienta hemos diseñado el modelo entidad/relación de nuestra base de datos.
- **PUTTY:** Es un emulador gratuito de terminal que soporta SSH y muchos otros protocolos, utilizado para conectarnos al servidor web (Linux) ubicado en la Raspberry a través de SSH.
- **FILEZILLA:** Gestor de archivos FTP, utilizado para subir los archivos de la web al servidor.

## 2.2 LENGUAJES UTILIZADOS

Los lenguajes de programación que hemos utilizado en la creación de la página web, así como, el acceso a la base de datos por parte de esta son los siguientes:

- **HTML:** No es como tal un lenguaje de programación, sino, un lenguaje de marcado que hemos utilizado para la elaboración de la página web.
- **CSS:** Hojas de estilo en cascada, es un lenguaje de diseño gráfico para definir y dar estilos a nuestro documento escrito en lenguaje de marcado, en HTML.
- **PHP:** Es un lenguaje de código abierto especialmente adecuado para el desarrollo web y que se puede incrustar fácilmente en HTML. En nuestro caso, sobre todo, lo hemos utilizado para extraer información de la base de datos de la empresa e interpretarla.

*Ejemplo de PHP en la web:*

```
<?php

session_start();
require_once "conexion.php";

$conexion=conexion();

$usuario=$_POST['usuario'];
$pass=sha1($_POST['password']);

$sql="SELECT * from usuarios where usuario='$usuario' and password='$pass'";
$result=mysqli_query($conexion,$sql);

if(mysqli_num_rows($result) > 0){
    $_SESSION['user']=$usuario;
    echo 1;
}else{
    echo 0;
}

?>
```

En este caso, siempre y cuando la conexión.php esté establecida, recogemos el valor de los input usuario y contraseña (con sha1 que es un algoritmo seguro para guardar contraseñas) y comparamos esta información con la base de datos, con *mysqli\_num\_rows* obtenemos el número de filas de resultados, y si es correcta, tendremos acceso a la zona de cliente.

- **JAVASCRIPT:** Es un lenguaje de programación compilado, orientado a objetos y se utiliza principalmente del lado del cliente, posibilitando la creación de páginas web dinámicas.

## Ejemplo de JAVASCRIPT en la web:

```
<script type="text/javascript">

var detectarNavegador = window.navigator.userAgent;
var chrome = /Chrome/;
var firefox = /Firefox/;
var opera = /OPR/;
var edge = /Edge/;
var safari = /Safari/;
var navegador;
if(chrome.test(detectarNavegador) && !(opera.test(detectarNavegador)) && !(edge.test(detectarNavegador))){
    navegador = 'Google Chrome';
}else if(firefox.test(detectarNavegador)){
    navegador = 'Firefox Mozilla';
    alert('Usted está utilizando: '+navegador +
    '\n Esta página está optimizada para Google Chrome');

}else if(safari.test(detectarNavegador) && !(chrome.test(detectarNavegador))){
    navegador = 'Apple Safari';
    alert('Usted está utilizando: '+navegador +
    '\n Esta página está optimizada para Google Chrome');
}else if(opera.test(detectarNavegador)){
    navegador = 'Opera';
    alert('Usted está utilizando: '+navegador +
    '\n Esta página está optimizada para Google Chrome');
}else if(edge.test(detectarNavegador)){
    navegador = 'Microsoft Edge';
    alert('Usted está utilizando: '+navegador +
    '\n Esta página está optimizada para Google Chrome');
}else{
    navegador = 'un navegador desconocido';
    alert('Usted está utilizando: '+navegador +
```

La página por el momento está optimizada únicamente para Chrome, utilizaremos JavaScript para, primero, detectar el navegador con el cual el cliente está accediendo a nuestra web y en el caso de que no sea Chrome, genere una alerta avisando que la web únicamente está optimizada para este navegador.

VPN localhost/Proyecto\_HOTEL\_RURAL/index.php

### localhost dice

Usted está utilizando: Opera  
Esta página está optimizada para Google Chrome

Aceptar

- **JQUERY:** Es una biblioteca multiplataforma de JavaScript, que permite simplificar la forma de desarrollar aplicaciones web, ya que se utiliza menos tiempo y menos código que las aplicaciones hechas con JavaScript puro.

*Ejemplo de jQuery en la web:*

```
<script src="http://code.jquery.com/jquery-latest.js"></script>
<script type="text/javascript">
    $(document).ready(function() {
        $('.booking').fadeIn(3500);
    });
</script>
```



En este caso, usando el efecto `fadeIn()`, nos muestra un `div` (`class=booking`), que anteriormente lo hemos establecido en `css` como `display:none`.

Este recuadro donde podremos realizar nuestra reserva se cargará con un retardo de 3500 milisegundos (3,5 segundos) con respecto al resto de elementos del `index.php`.

- **BOOTSTRAP:** Es un framework, o librería, que contiene plantillas de diseño que podemos utilizar para ahorrar tiempo en el diseño de nuestra web.

*Ejemplo de Bootstrap en la web:*

```
<html>
<head>
<link href="//netdna.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap.min.css" rel="stylesheet"

<div id="carousel-id" class="carousel slide" data-ride="carousel">
  <ol class="carousel-indicators">
    <li data-target="#carousel-id" data-slide-to="0" class="active"></li>
    <li data-target="#carousel-id" data-slide-to="1" class=""></li>
    <li data-target="#carousel-id" data-slide-to="2" class=""></li>
    <li data-target="#carousel-id" data-slide-to="3" class=""></li>
  </ol>
  <!-- FIRST CAROUSEL -->
  <div class="carousel-inner">
    <div class="item active">
```



Se usó en la página (index.php) para realizar un slider web (múltiples imágenes que se alternan entre ellas).

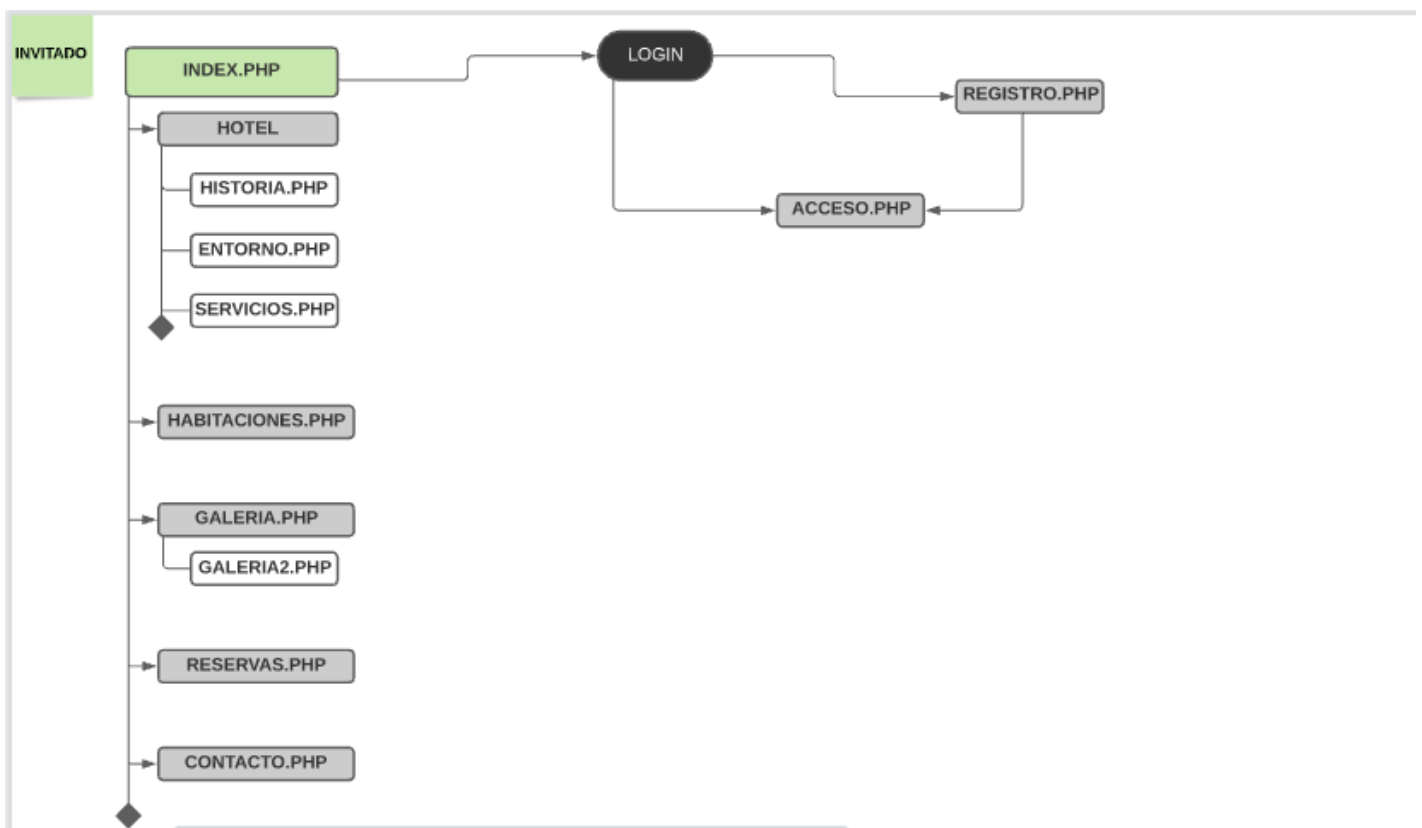
Hay dos formas de usar BOOTSTRAP, una es descargando el archivo de la página oficial e incluirlo en nuestro proyecto y la otra enlazarlo directamente (con un link) en la cabecera de nuestro HTML (head).

## 2.3 DISEÑO ARQUITECTÓNICO

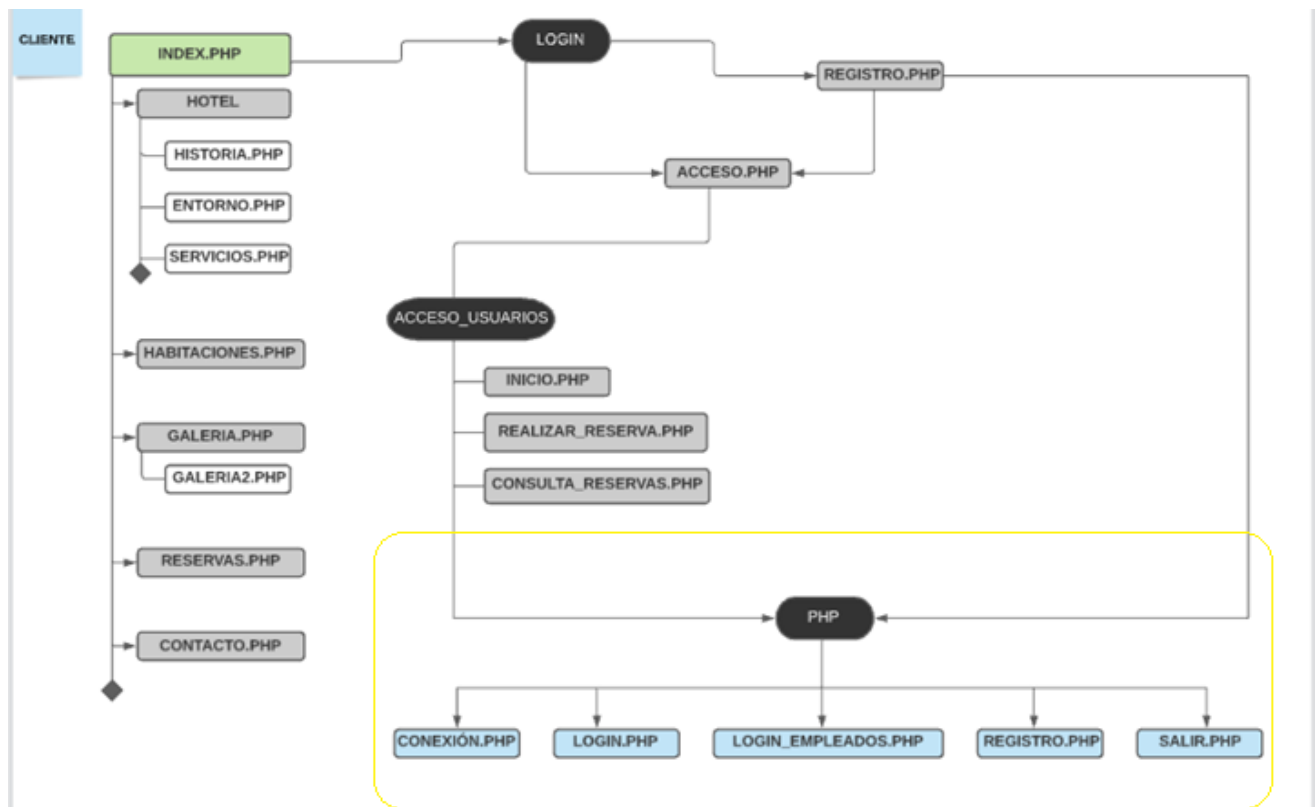
Se ha realizado el diseño arquitectónico del sitio web con la aplicación [Lucid App](#).

Podemos acceder a la web de tres modos:

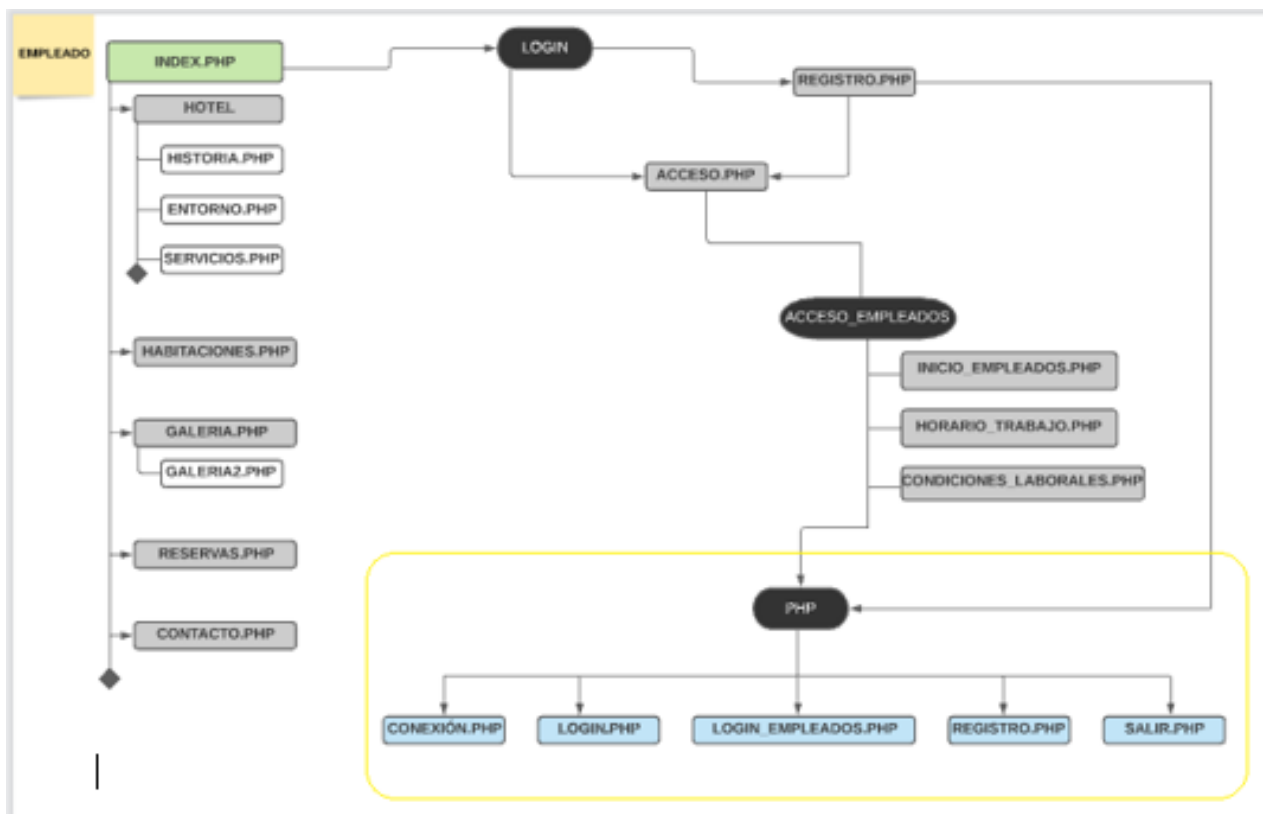
1) **Acceso Usuario Sin Identificar / Invitado:** Los usuarios invitados que no tengan cuenta en nuestro sitio web, podrán navegar por la página, pero no podrán acceder a cuentas de usuario ni realizar/consultar sus reservas.



2) **Acceso Cliente:** Los clientes podrán navegar por toda la web igualmente y podrán realizar y consultar las reservas realizadas en la misma:

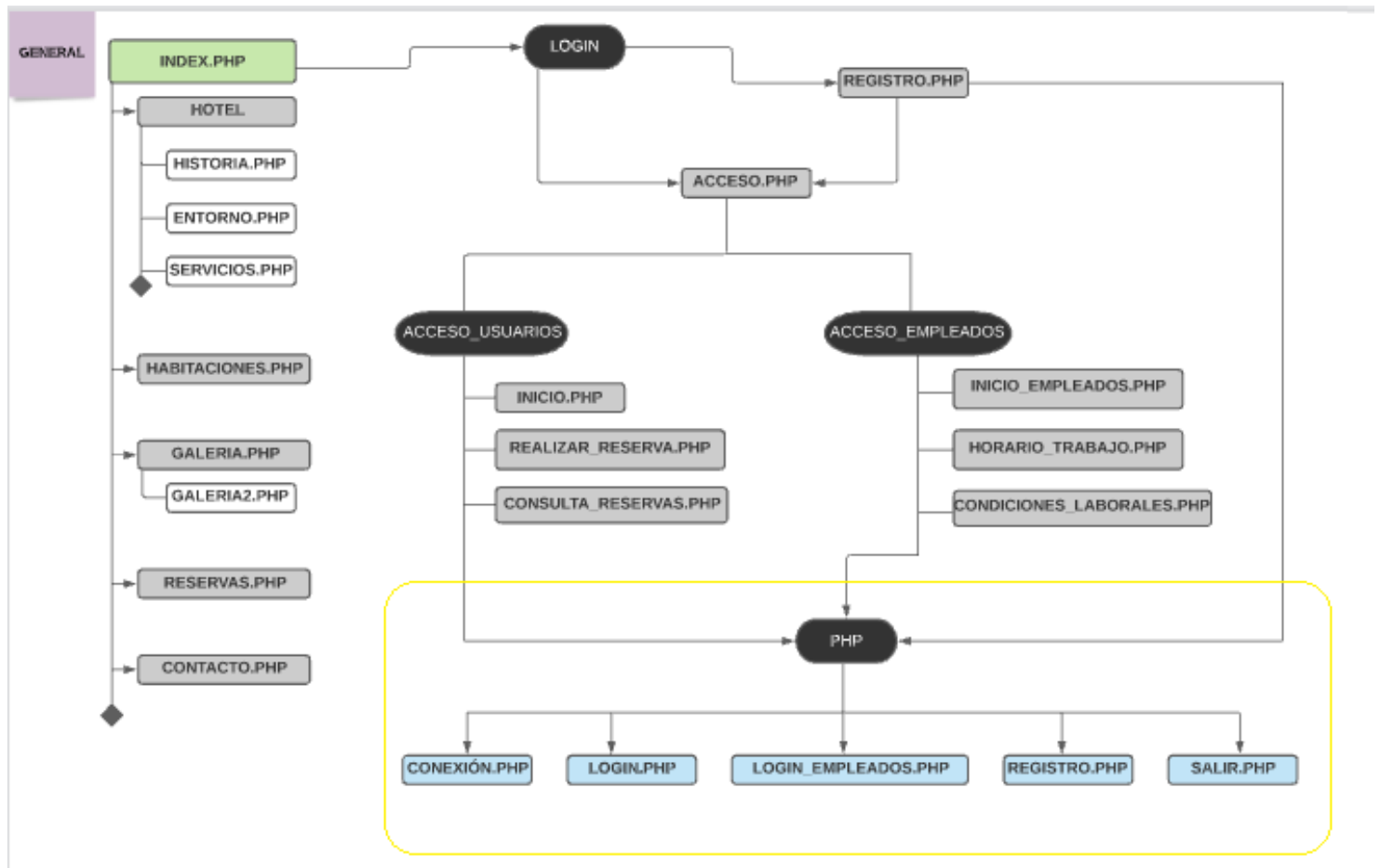


3) **Acceso Empleados:** En la cuenta de usuario de los trabajadores, se podrán consultar los horarios de trabajo, sus datos, etc.



Se marca con un recuadro amarillo aquellos archivos php fundamentales para la comprobación y validación de los datos a la hora del registro o acceso.

En resumen, el esquema general de la página sería el siguiente:

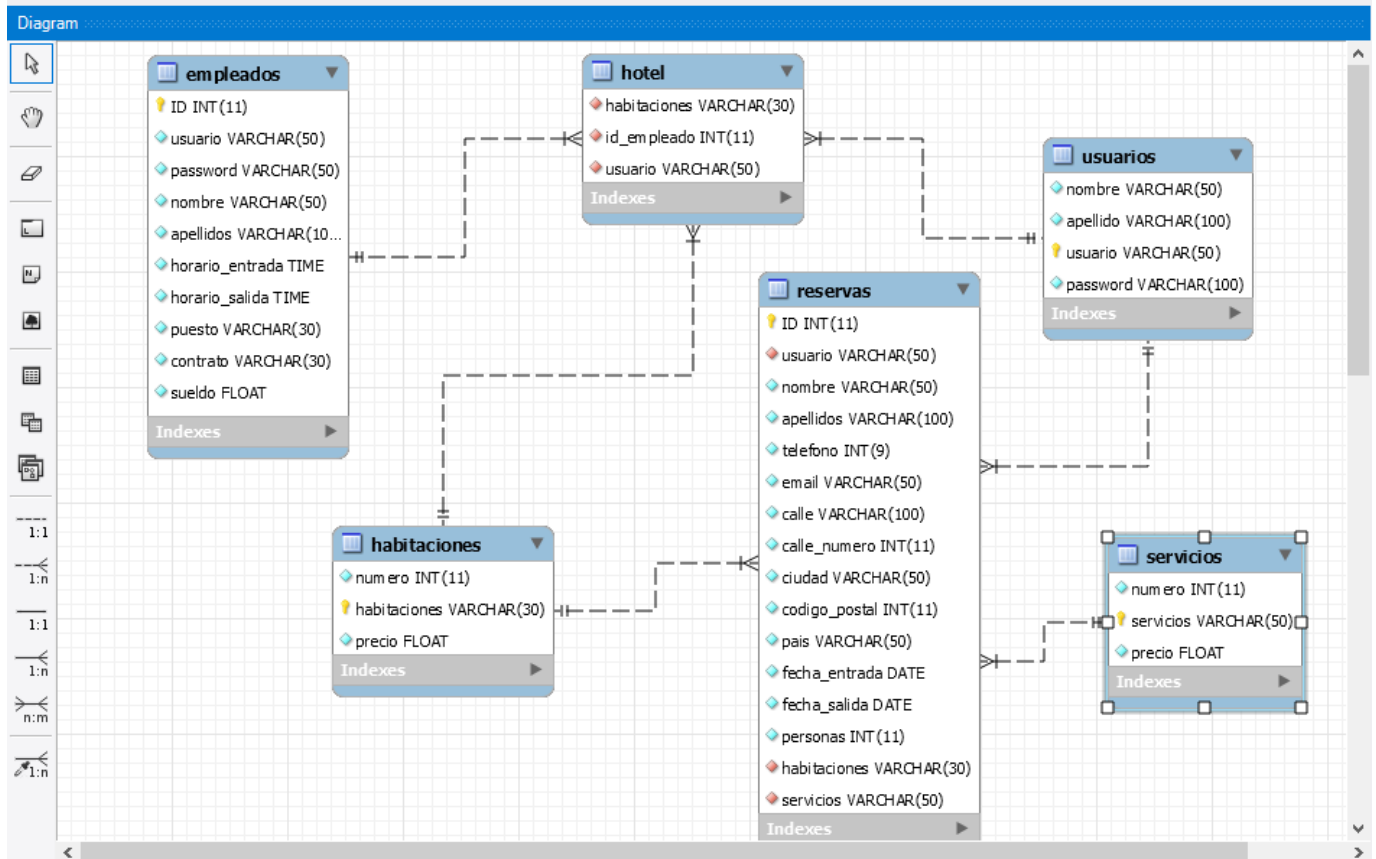


### 3. BASE DE DATOS

#### 3.1 DISEÑO DE BASE DE DATOS

El proyecto está enfocado en la creación y diseño de una página web, por lo tanto, se requiere de una base de datos para almacenar toda la información relativa a la página web y la empresa.

Consta de 6 tablas que se relacionan entre sí (empleados - hotel (n - 1); hotel - usuarios (1 - n); hotel - habitaciones (1 - n); usuarios - reservas (1 - n); habitaciones - reservas (1 - n); reservas - servicios (1 - 1)).



### 3.2 DEFINICIÓN DE TABLAS

**TABLA EMPLEADOS:** Recoge a los trabajadores de la empresa. La clave principal de la tabla es ID, que además es un campo auto incremental.

```
-- Estructura de tabla para la tabla `empleados`
CREATE TABLE `empleados` (
  `ID` int(11) NOT NULL primary key auto_increment,
  `usuario` varchar(50) NOT NULL,
  `password` varchar(50) NOT NULL,
  `nombre` varchar(50) NOT NULL,
  `apellidos` varchar(100) NOT NULL,
  `horario_entrada` time NOT NULL,
  `horario_salida` time NOT NULL,
  `puesto` varchar(30) NOT NULL,
  `contrato` varchar(30) NOT NULL,
  `sueldo` float NOT NULL
)
```

Insertamos manualmente los datos de nuestros trabajadores, proporcionándoles a cada uno en privado su contraseña de acceso.

```
1 INSERT INTO `empleados` (`ID`, `usuario`, `password`, `nombre`, `apellidos`, `horario_entrada`,
  `horario_salida`, `puesto`, `contrato`, `sueldo`) VALUES
2 (1, 'V123456', SHA1('7110eda'), 'Julian', 'López', '15:00:00', '21:00:00', 'Conserjería', 'Indefinido', 1000),
3 (2, 'V223456', SHA1('0d2c0220'), 'Ana', 'García', '09:00:00', '17:00:00', 'Técnico Jurídico Laboral',
  'Indefinido', 1450),
4 (3, 'V323456', SHA1('062aa5e4'), 'Luis', 'Gómez', '09:00:00', '17:00:00', 'Técnico Contable', 'Indefinido',
  1450),
5 (4, 'V423456', SHA1('0b0a572ac0d2c0220'), 'Carlos', 'Gutiérrez', '09:00:00', '17:00:00', 'Community Manager',
```

En la siguiente imagen vemos como quedan las contraseñas después de usar el cifrado SHA1

ID	usuario	password	nombre	apellidos	horario_entrada	ho
1	V123456	7110eda	Julian	López	15:00:00	21:
2	V223456	0d2c0220	Ana	García	09:00:00	17:
3	V323456	062aa5e4	Luis	Gómez	09:00:00	17:
4	V423456	0b0a572ac0d2c0220	Carlos	Gutiérrez	09:00:00	17:
5	V523456	a4d09e0	Julio	Álvarez	09:00:00	17:
6	V623456	7110eda	Rosa	Benitez	09:00:00	17:
7	V723456	110da4a4a390b0a572	Paula	Benito	17:00:00	22:
8	V823456	390	Miguel	Villanueva	09:00:00	17:

**TABLA HABITACIONES:** En ella registramos el tipo de habitación que tenemos disponible (Individual, Doble o Suite) y el precio de cada una de ellas. **Habitaciones** es la primary key de la tabla.

```
-- Estructura de tabla para la tabla `habitaciones`

• CREATE TABLE `habitaciones` (
  `numero` int(11) NOT NULL,
  `habitaciones` varchar(30) NOT NULL primary key,
  `precio` float NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

• INSERT INTO `habitaciones` (`numero`, `habitaciones`, `precio`) VALUES
(2, 'Doble', 50),
(1, 'Individual', 40),
(3, 'Suite', 100);
```

**TABLA RESERVAS:** Almacena todas las reservas realizadas en la web. Esta tabla contiene tres keys: primary key (ID) y dos foreign key (Habitaciones y servicios):

```
-- Estructura de tabla para la tabla `reservas`  
  
CREATE TABLE `reservas` (  
  `ID` int(11) NOT NULL PRIMARY KEY auto_increment,  
  `usuario` varchar(50) NOT NULL,  
  `nombre` varchar(50) NOT NULL,  
  `apellidos` varchar(100) NOT NULL,  
  `telefono` int(9) NOT NULL,  
  `email` varchar(50) NOT NULL,  
  `calle` varchar(100) NOT NULL,  
  `calle_numero` int(11) NOT NULL,  
  `ciudad` varchar(50) NOT NULL,  
  `codigo_postal` int(11) NOT NULL,  
  `pais` varchar(50) NOT NULL,  
  `fecha_entrada` date NOT NULL,  
  `fecha_salida` date NOT NULL,  
  `personas` int(11) NOT NULL,  
  `habitaciones` varchar(30) NOT NULL,  
  `servicios` varchar(50) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

**TABLA HOTEL:** Esta tabla sirve de nexo entre las tablas habitaciones-empleados y usuarios.

```
-- Estructura de tabla para la tabla `hotel`  
  
CREATE TABLE `hotel` (  
  `habitaciones` varchar(30) NOT NULL,  
  `id_empleado` int(11) NOT NULL,  
  `usuario` varchar(50) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

**TABLA SERVICIOS:** Almacena los servicios que oferta el Hotel, como primary key se establece **servicios**, que a la vez es foreign key de la tabla reservas.

```
-- Estructura de tabla para la tabla `servicios`
```

```
CREATE TABLE `servicios` (  
  `numero` int(11) NOT NULL,  
  `servicios` varchar(50) NOT NULL primary KEY,  
  `precio` float NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
INSERT INTO `servicios` (`numero`, `servicios`, `precio`) VALUES  
(2, 'Escalada', 25),  
(3, 'Paintball', 50),  
(1, 'Rutas', 10);
```

**TABLA USUARIOS:** Cuando un usuario se registra en nuestra web, automáticamente se inserta en esta tabla, tiene el campo **usuario** como primary key y la recogida de datos en el campo password con el algoritmo de cifrado SHA1.

```
-- Estructura de tabla para la tabla `usuarios`
```

```
CREATE TABLE `usuarios` (  
  `nombre` varchar(50) NOT NULL,  
  `apellido` varchar(100) NOT NULL,  
  `usuario` varchar(50) NOT NULL PRIMARY KEY,  
  `password` varchar(100) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

### 3.3 RELACIONES

Para realizar las relaciones entre las tablas, lo primero es declarar las Keys (tabla hotel: habitaciones, id empleado y usuario; tabla reservas: usuario, habitaciones y servicios)

```
1  -- Indices de la tabla `hotel`  
2  • ALTER TABLE `hotel`  
3      ADD KEY `habitaciones` (`habitaciones`),  
4      ADD KEY `id_empleado` (`id_empleado`),  
5      ADD KEY `usuario` (`usuario`);  
6  
7  -- Indices de la tabla `reservas`  
8  • ALTER TABLE `reservas`  
9      ADD KEY `usuario` (`usuario`),  
10     ADD KEY `habitaciones` (`habitaciones`),  
11     ADD KEY `servicios` (`servicios`);  
12
```

Y, a continuación, construimos la relación entre las keys declaradas y las claves referenciadas:

```
-- Filtros para la tabla `hotel`  
• ALTER TABLE `hotel`  
  ADD CONSTRAINT `hotel_ibfk_1` FOREIGN KEY (`id_empleado`) REFERENCES `empleados` (`ID`),  
  ADD CONSTRAINT `hotel_ibfk_2` FOREIGN KEY (`habitaciones`) REFERENCES `habitaciones` (`habitaciones`),  
  ADD CONSTRAINT `hotel_ibfk_3` FOREIGN KEY (`usuario`) REFERENCES `usuarios` (`usuario`);  
-- Filtros para la tabla `reservas`  
• ALTER TABLE `reservas`  
  ADD CONSTRAINT `reservas_ibfk_1` FOREIGN KEY (`habitaciones`) REFERENCES `habitaciones` (`habitaciones`),  
  ADD CONSTRAINT `reservas_ibfk_2` FOREIGN KEY (`usuario`) REFERENCES `usuarios` (`usuario`),  
  ADD CONSTRAINT `reservas_ibfk_3` FOREIGN KEY (`servicios`) REFERENCES `servicios` (`servicios`);
```

### 3.4 CONEXIÓN CON LA BBDD

Para realizar la conexión con la base de datos tenemos que usar el lenguaje PHP para conectarnos con la base de datos y así poder obtener resultados de esta:

*(Las siguientes imágenes de código PHP las hemos realizado en la web [Carbon](#) que reescribe tu código de forma que quede lo más claro posible)*

**Función conexión()** nos retorna el valor de la variable \$conexion que es la unión con la base de datos

```
<?php  
function conexion()  
{  
    return $conexion=mysqli_connect("localhost","root","XXXXXXXXXX","hotelSierraDeGata");  
}
```



**Recogida de datos de los input del formulario de registro:** en cada página de conexión php, debemos colocar en el inicio la frase “**require\_once** “**conexion.php**””, de modo que, si no lograse conectarse con la base de datos no establezca la conexión.

Recogemos los valores del input password con el algoritmo de cifrado **SHA1**

```
<?php

require_once "conexion.php";
$conexion=conexion();

$nombre=$_POST['nombre'];
$apellido=$_POST['apellido'];
$usuario=$_POST['usuario'];
$password=sha1($_POST['password']);

if(buscaRepetido($usuario,$conexion)==1){
    echo 2;
}else{
    $sql="INSERT into usuarios (nombre,apellido,usuario,password)
    values ('$nombre','$apellido','$usuario','$password')";
    echo $result=mysqli_query($conexion,$sql);
}

function buscaRepetido($user,$conexion){
    $sql="SELECT * from usuarios
    where usuario='$user'";
    $result=mysqli_query($conexion,$sql);

    if(mysqli_num_rows($result) > 0){
        return 1;
    }else{
        return 0;
    }
}
```

Para poder comprobar si el nombre de un usuario ya existe a la hora de registrarse, creamos la función php **buscaRepetido**, y con **mysqli\_num\_rows** — obtenemos el número de filas de un resultado, si es mayor que 0 significa que ese usuario ya existe y debemos elegir otro.

**Insertar datos en la tabla reservas:** cuando realizamos una reserva en la cuenta de usuario:

```
<?php
require_once "conexion.php";
$conexion=conexion();
session_start();

$usuario= ($_SESSION['user']);
$nombre=$_POST['nombre'];
$apellidos=$_POST['apellidos'];
$telefono=$_POST['telefono'];
$email=$_POST['email'];
$calle=$_POST['calle'];
$calle_numero=$_POST['calle_numero'];
$ciudad=$_POST['ciudad'];
$codigo_postal=$_POST['codigo_postal'];
$pais=$_POST['pais'];
$fecha_entrada=$_POST['fecha_entrada'];
$fecha_salida=$_POST['fecha_salida'];
$personas=$_POST['personas'];
$habitaciones=$_POST['habitaciones'];
$servicios=$_POST['servicios'];

$sql = "INSERT INTO reservas (usuario,nombre,apellidos,telefono,email,calle,calle_numero,
ciudad,codigo_postal,pais,fecha_entrada,fecha_salida,personas,habitaciones,servicios)
VALUES ('$usuario','$nombre','$apellidos','$telefono','$email','$calle','$calle_numero',
'$ciudad','$codigo_postal','$pais','$fecha_entrada','$fecha_salida','$personas','$habitaciones','$servi
cios')";

if (mysqli_query($conexion, $sql)) {
    echo'<script type="text/javascript">
        alert("Se ha registrado correctamente su reserva, nos pondremos en contacto a la mayor
brevedad");
        window.location.href="inicio.php";
    </script>';
} else {
    echo "Error: " . $sql . " " . mysqli_error($conexion);
}
mysqli_close($conexion);
?>
```

### **Página inicio sesión Usuarios/Empleados:**

En cada una de ellas, debemos hacer una sentencia condicional con la sesión iniciada del usuario al inicio del documento:

```
<?php
session_start();

if(isset($_SESSION['user'])){

?>
```

```

<?php
    $usuario=$_SESSION['user'];
    $bd=new mysqli();
    $bd->connect('localhost','root','[REDACTED]','hotelsierradegata');
    $error=$bd->connect_errno;

    if ($error!=null)
    {
        <?php
        print "Error " . $error . " al conectar con la base de datos";
        ?>
        <?php
    }
    $resultado=$bd->query("Select * from empleados where usuario='$usuario'");
    $datos=$resultado->fetch_array();
    $nom=$datos[3];
    $apell=$datos[4];

    ?>

```

Realizamos un array con el resultado que queremos enseñar de nuestra base de datos y entonces, lo mostramos en la página mediante \$datos[3] (el número que nos interese mostrar, por ejemplo: [0] el nombre, [1] apellidos, etc....

Este código es válido tanto para realizar\_reservas.php, como consultar\_reservas.php y para horario\_trabajadores.php y condiciones\_laborales.ph

**Salir de la sesión y base de datos:**

```

<?php
    session_start();

    unset($_SESSION['user']);

    header("location:../../index.php");

    ?>

```

## 4. ARQUITECTURA DE RED

### 4.1 ESQUEMA Y ESTRUCTURA DE LA RED

El esquema principal de la red:

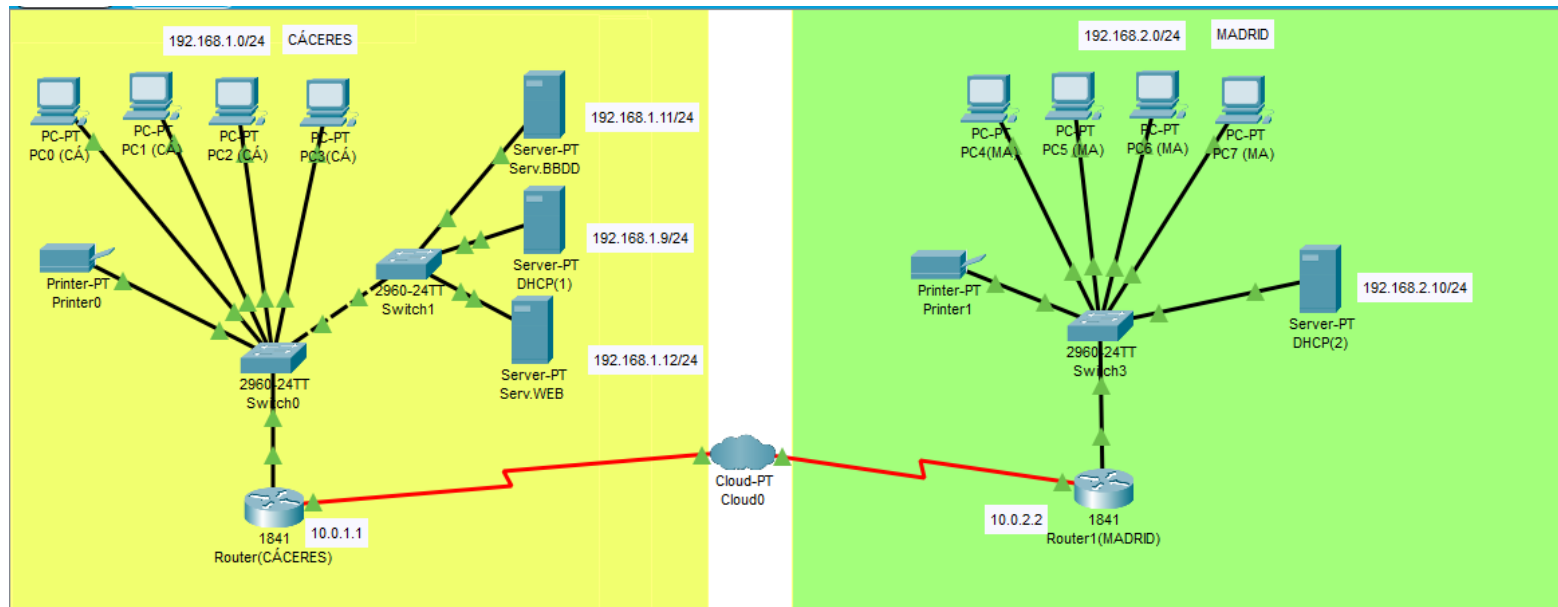


Tabla con direcciones de todos los dispositivos:

Dispositivo	Dirección IP	Máscara de red	Puerta de Enlace
PC0 (Cáceres)	192.168.1.4	255.255.255.0	192.168.1.1
PC1	192.168.1.5	255.255.255.0	192.168.1.1
PC2	192.168.1.6	255.255.255.0	192.168.1.1
PC3	192.168.1.7	255.255.255.0	192.168.1.1
Printer0	192.168.1.10	255.255.255.0	192.168.1.1
Switch0			
Switch1			
Serv.BBDD	192.168.1.11	255.255.255.0	192.168.1.1
Serv.DHCP(1)	192.168.1.9	255.255.255.0	192.168.1.1
Serv.WEB	192.168.1.12	255.255.255.0	192.168.1.1
Router Cáceres	192.168.1.1	255.255.255.0	
	10.0.1.1	255.0.0.0	

Dispositivo	Dirección IP	Máscara de red	Puerta de Enlace
PC4(Madrid)	192.168.2.11	255.255.255.0	192.168.2.1
PC5	192.168.2.12	255.255.255.0	192.168.2.1
PC6	192.168.2.15	255.255.255.0	192.168.2.1
PC7	192.168.2.13	255.255.255.0	192.168.2.1
Printer1	192.168.2.14	255.255.255.0	192.168.2.1
Switch3			
DHCP(2)	192.168.2.10	255.255.255.0	192.168.2.1
Router Madrid	192.168.2.1	255.255.255.0	
	10.0.2.2	255.0.0.0	

La configuración de los servidores DHCP que van a repartir las IP's de ambas oficinas es la siguiente:

DHCP 1: Reparte direcciones desde 192.168.1.0.

Pool Name	Default Gateway	DNS Server	Start IP Address	Subnet Mask	Max User	TFTP Server	WLC Address
Ip's Cáceres	192.168.1.1	0.0.0.0	192.168.1.0	255.255.255.0	246	0.0.0.0	0.0.0.0

DHCP 2: Reparte direcciones desde 192.168.2.10.

Pool Name	Default Gateway	DNS Server	Start IP Address	Subnet Mask	Max User	TFTP Server	WLC Address
Ip's Madrid	192.168.2.1	0.0.0.0	192.168.2.10	255.255.255.0	246	0.0.0.0	0.0.0.0

Una vez diseñado el esquema de nuestra red, vamos a comprobar que dentro de la misma tenemos conexión, haciendo ping o enviando un paquete al router:

Desde PC0 (Cáceres) al Router (Cáceres):

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Pe
	Successful	PC0 (CÁ)	Router(CÁCERES)	ICMP		0.000	
	Successful	Router(CÁCERES)	PC0 (CÁ)	ICMP		0.000	

Desde PC4 (Madrid) al Router (Madrid)

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Pe
	Successful	PC4(MA)	Router1(MADRID)	ICMP		0.000	
	Successful	Router1(MADRID)	PC4(MA)	ICMP		0.000	

Comprobamos también, la conexión entre diferentes dispositivos de la misma red.

Conexión PC's Red Cáceres con Servidores e Impresoras

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Pe
	Successful	PC0 (CÁ)	Serv.BBDD	ICMP		0.000	
	Successful	Serv.BBDD	PC0 (CÁ)	ICMP		0.000	

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Pe
	Successful	PC3(CÁ)	Serv.WEB	ICMP		0.000	
	Successful	Serv.WEB	PC3(CÁ)	ICMP		0.000	

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num
	Successful	PC2 (CÁ)	Printer0	ICMP		0.000	N	0
	Successful	PC6 (MA)	Printer1	ICMP		0.000	N	1

### Conexión entre servidores

Fire	Last Status	Source	Destination	Type	Color	Time(sec)
	Successful	Serv.BBDD	Serv.WEB	ICMP		0.000
	Successful	DHCP(1)	Serv.WEB	ICMP		0.000
	Successful	DHCP(1)	Serv.BBDD	ICMP		0.000
	Successful	Serv.WEB	Serv.BBDD	ICMP		0.000

## 4.2 CONEXIÓN ENTRE OFICINAS/REDES

Hemos conectado las dos sedes gracias a la nube con la opción **Frame Relay**, que es una tecnología de protocolo de red de conmutación de paquetes digital de capa de enlace de datos diseñada para conectar **redes de área local (LAN)** y transferir datos a través de **redes de área amplia (WAN)**.

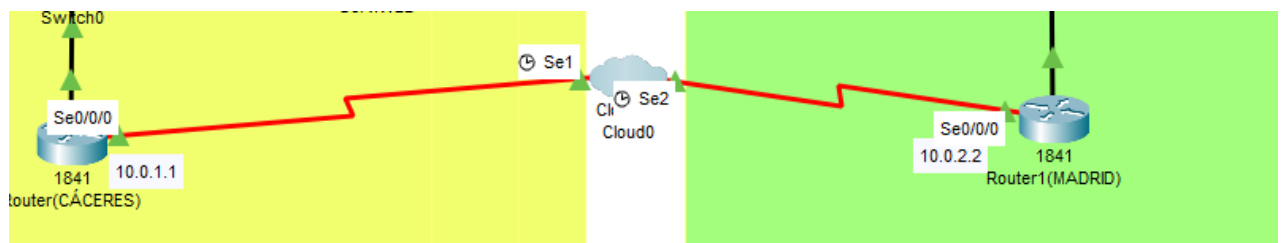
Para ello, inicialmente otorgamos al router su dirección IP y dirección para el serial:

Router0 (Cáceres):

- Ethernet: 192.168.1.1/24
- Serial: 10.0.1.1

Router1 (Madrid):

- Ethernet: 192.168.1.2/24
- Serial: 10.0.2.2

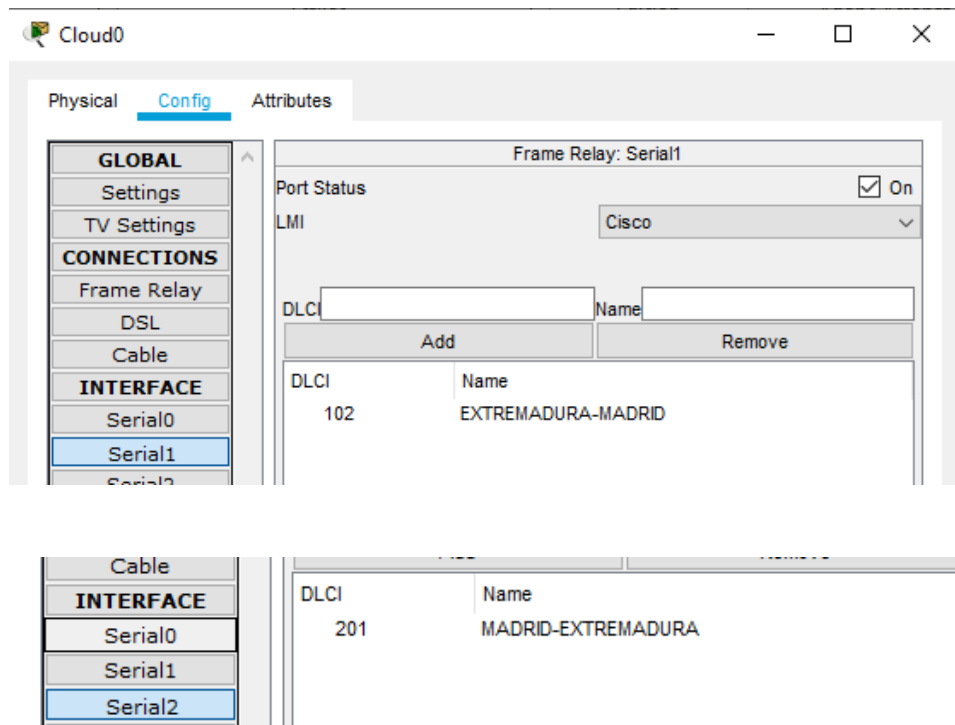


El siguiente paso es configurar el DLCI de la nube

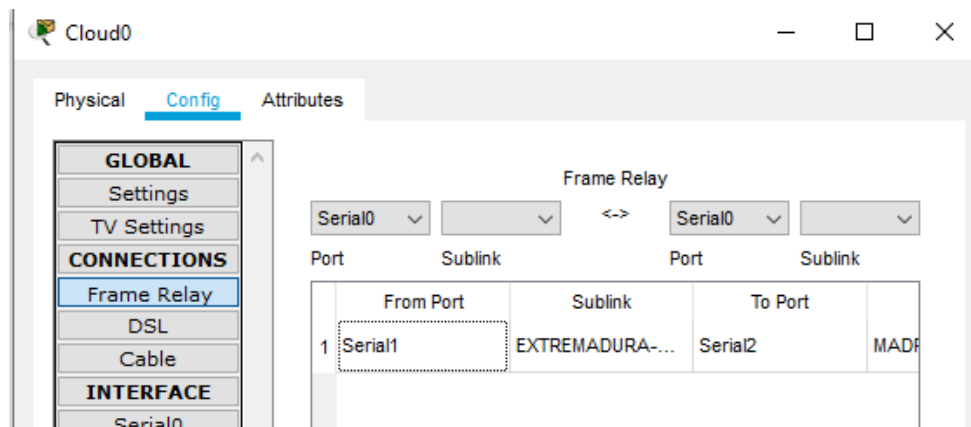
DLCI es el identificador de conexión del circuito establecido en Frame Relay. Este identificador se aloja en la trama e indica el camino a seguir por los datos, es decir, el circuito virtual establecido.

Queremos conectar los routers de Cáceres y Madrid .

Para ello abrimos la nube y en DLCI, conectaremos el 1(Cáceres) 0 con el 2(Madrid) y viceversa:



Y en la pestaña “Frame Realay” unimos ambos serial:



Para terminar de configurarlo, debemos indicar (por CLI) que establezca la conexión, en ambos routers escribiremos “encapsulation frame-relay”

```
Router(config-if)# encapsulation frame-relay
Router(config-if)#
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed
state to up
```

Antes de comprobar que los paquetes pueden llegar desde una oficina a la otra, debemos configurar nuestro router por **encaminamiento estático**, es decir, añadir al router las direcciones de red, él mismo interpretará si ese mensaje es para la red a la que pertenece, y si es así, lo enviará al dispositivo que espera ese paquete.

#### ROUTER CÁCERES:

#### ROUTER MADRID:

A continuación probamos si funciona el Frame Relay y hay conexión entre dispositivos de la red Cáceres con la Red Madrid:

#### Conexión entre routers:

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Pe
	Successful	Router1(MADRID)	Router(CÁCERES)	ICMP		0.000	
	Successful	Router(CÁCERES)	Router1(MADRID)	ICMP		0.000	

#### Conexión PC's Red Madrid a PC's Cáceres/Servidores Red Cáceres:

PC4 (Red Madrid) a Servidor BBDD (Red Cáceres)

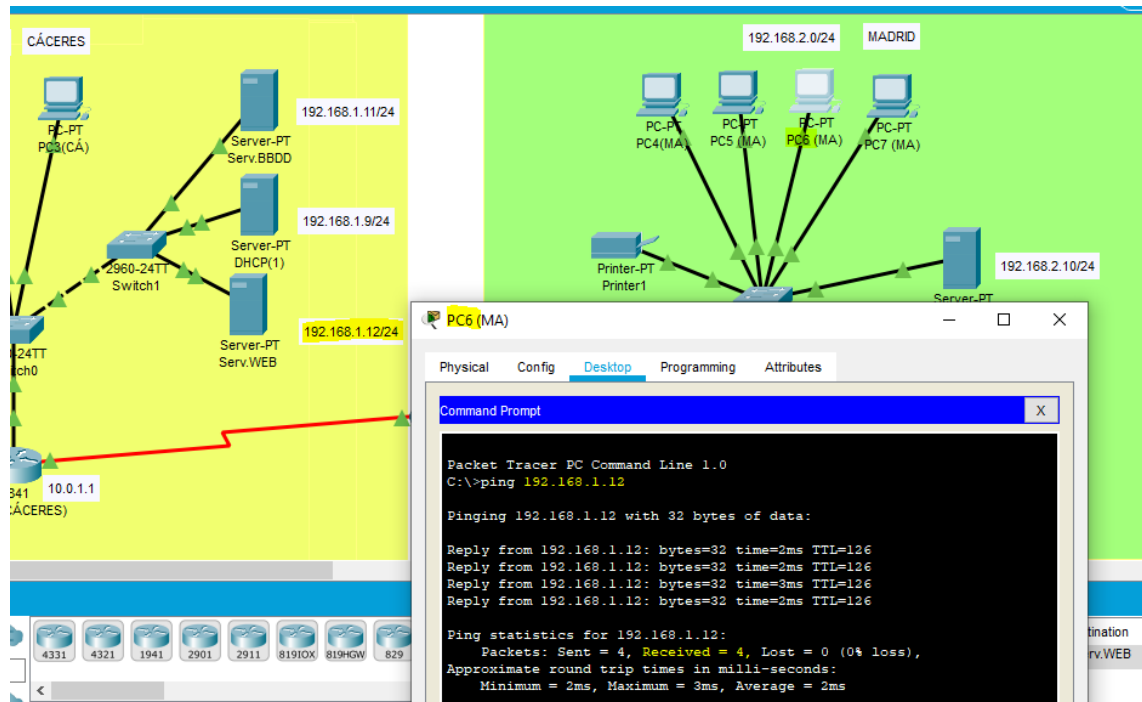
Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Pe
	Successful	Serv.BBDD	PC4(MA)	ICMP		0.000	
	Successful	PC4(MA)	Serv.BBDD	ICMP		0.000	



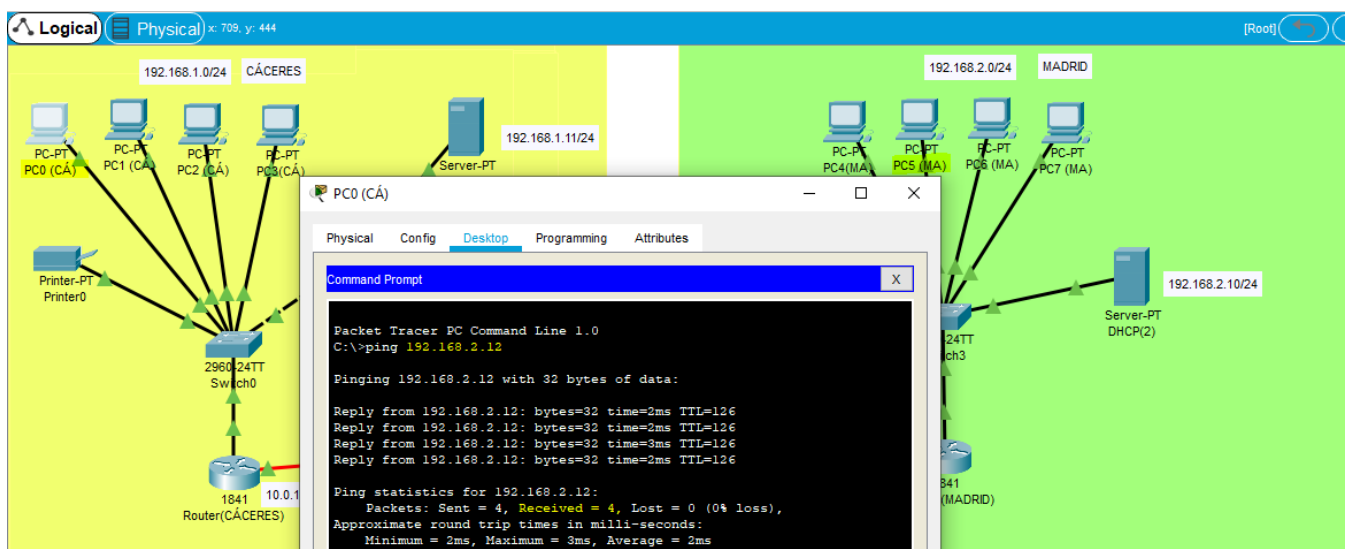
## PC7 (Madrid) a PC1 (Cáceres)

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Pe
	Successful	PC7 (MA)	PC1 (CÁ)	ICMP		0.000	
	Successful	PC1 (CÁ)	PC7 (MA)	ICMP		0.000	

## Conexión por ping - CLI entre PC6 (Madrid) y Serv.Web (Cáceres -192.168.1.12)



## Desde PC0 (Cáceres) a PC5 (Madrid – 192.168.2.12)



## 5. DISEÑO WEB

### 5.1 SECCIONES WEB

#### 5.1.1 INDEX

Index.php, es la página principal de nuestro sitio web.

En ella hemos querido informar sobre el hotel, su localización, fotos de este y la posibilidad de reservar una habitación.

El Header se compone de:

-Logo de la empresa:



-Redes sociales



```
<!-- RSS -->
<div style="float:left;">
  <a href="facebook.com" class="fa fa-facebook"></a>
  <a href="twitter.com" class="fa fa-twitter"></a>
  <a href="instagram.com" class="fa fa-instagram"></a>
</div>
```

-Lista de navegación

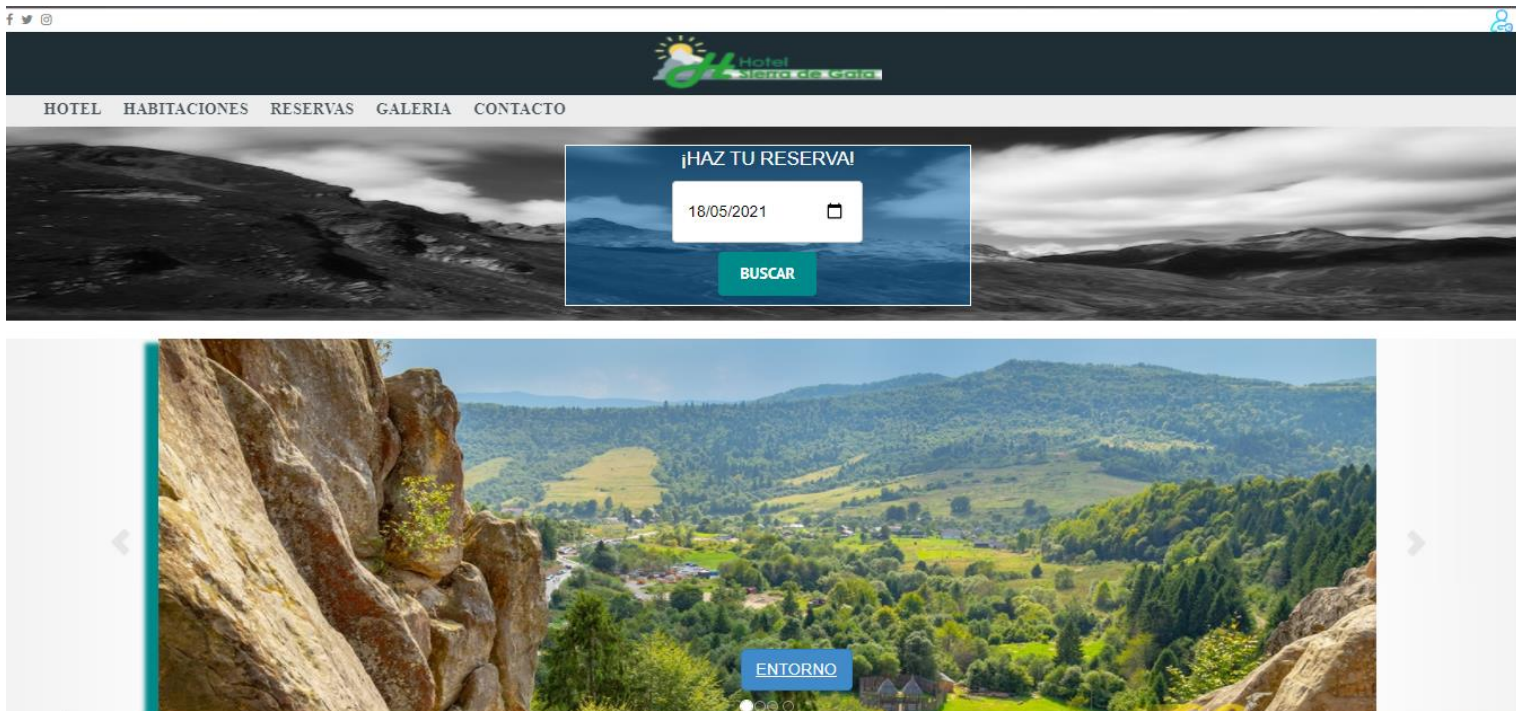


El diseño aplicado a la lista navegación es el siguiente:

```
#navigation>li {
    float:left;
    display:block;
    list-style-type:none;
}
#navigation a {
    display:block;
    padding:2px 2px;
    color:#1F2E34;
    font-weight: bold;
    height:33px;
    text-align: center;
}
#navigation a:hover {
    border-left:4px solid white;
    border-right:4px solid white;
    transform: scale(1.1);
    color:#1F2E34;
    font-weight:bold;
    background-color: #EDED;
    text-decoration: underline;
}
```

### -Div “Haz Tu Reserva”:

Si buscamos una fecha y pulsamos en buscar, nos redirecciona a acceso.php para gestionarlo desde nuestra cuenta. ( jQuery – Fadein() )



Continuando en la página principal, nos encontraremos con una breve descripción del hotel y de sus servicios

## Hotel Rural Sierra de Gata

El hotel Rural Sierra de Gata es un hotel situado en pleno corazón de la Sierra de Gata, en el noroeste de la provincia de Cáceres. Limita al norte con la provincia de Salamanca, al oeste con Portugal y al este con las comarcas de Las Hurdes.

Te sorprenderá el paisaje, un espacio natural ideal para relajarse o para disfrutar de múltiples actividades.

Wifi gratuito en todo el establecimiento

Parking Gratuito

Zona de Gimnasio disponible

Reserva Online 100% Segura

De nuevo usando jQuery y la función *FadeIn()*, hemos establecido que los cuatro servicios (wifi,parking,gimnasio,reserva) vayan apareciendo sucesivamente con una diferencia de unos pequeños segundos:

```
<script src="http://code.jquery.com/jquery-latest.js"></script>
<script type="text/javascript">
    $(document).ready(function() {

        $('#wifi_widget').fadeIn(8500);
        $('#parking_widget').fadeIn(9500);
        $('#gym_widget').fadeIn(10500);
        $('#reserva_widget').fadeIn(11000);
    });
</script>
```

## Hotel Rural Sierra de Gata

El hotel Rural Sierra de Gata es un hotel situado en pleno corazón de la Sierra de Gata, en el noroeste de la provincia de Cáceres. Limita al norte con la provincia de Salamanca, al oeste con Portugal y al este con las comarcas de Las Hurdes.

Te sorprenderá el paisaje, un espacio natural ideal para relajarse o para disfrutar de múltiples actividades.

Wifi gratuito en todo el establecimiento

Parking Gratuito

Zona de Gimnasio disponible

Reserva Online 100% Segura

Antes de llegar a la parte final de la página principal (footer), encontramos una sección donde hemos insertado un mapa de Google Maps indicándonos la dirección de nuestro hotel:

```
<!--MAPS-->
```

```
<iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d390422.32073143416!2d-6.970"
```



## 5.1.2 HABITACIONES - SERVICIOS

En el caso de que se quiera reservar una habitación o un servicio, un cuadro de texto nos advertirá de que debemos iniciar sesión para realizar estos trámites.

### HABITACIONES

#### Habitación Individual



Desde 40.00€

Habitación Individual

RESERVAR

#### Habitación Doble



Desde 50.00€

Habitación Doble

RESERVAR



#### Suite



Desde 100.00€

Suite

RESERVAR



## SERVICIOS



### RUTAS

Entorno fluvial – Robledillo de Gata  
Embalse de Borbollón – Santibáñez El Alto  
Penamacor-Portugal

Desde **10.00€**

RESERVAR



### ESCALADA

Escalada Alange - Badajoz.  
Escalada Sierra del Castellar - Zafra.  
Escalada La Roca de la Sierra - Badajoz.

Desde **25.00€**

RESERVAR



### PAINTBALL

Campo de paintball situado en Sierra de Fuentes.  
Paintball infantil 7-14 años.  
Paintball adulto +14 años.

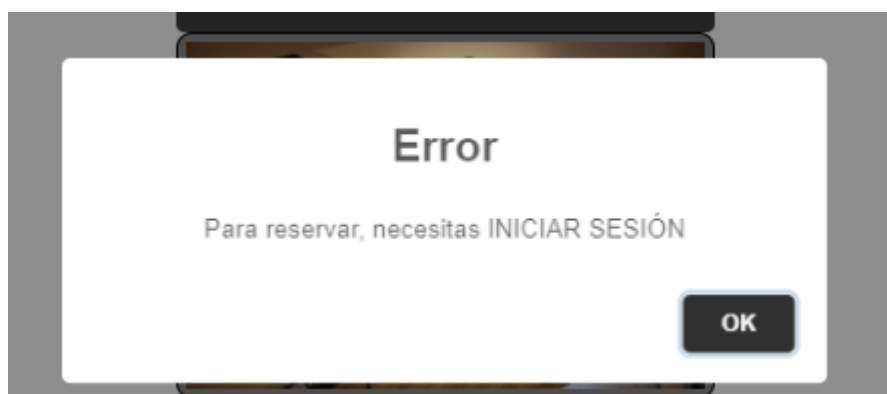
Desde **50.00€**

RESERVAR

Para realizar la sombra de color azul que aparece en las imágenes al pasar el ratón por encima, hemos usado la clase *hover* y la propiedad *transform*:

```
#first_room {  
  -webkit-transform: scale(1);  
  transform: scale(1);  
  -webkit-transition: .3s ease-in-out;  
  transition: .0s ease-in-out;  
}  
  
#first_room: hover {  
  -webkit-transform: scale(1.3);  
  transform: scale(1.1);  
  -webkit-transition: .4s ease-in-out;  
  box-shadow: 10px 30px 20px -5px darkcyan;  
}
```

Si clicamos en cualquier habitación o servicio a reservar aparecerá un cuadro emergente:



Si clicamos en OK y cerramos el diálogo, nos redireccionará a la página 'acceso.php' para loguearnos en nuestra cuenta y realizar/consultar las reservas.

### 5.1.3 CONTACTO

Un formulario de contacto sencillo (se está trabajando actualmente en un sistema de envío de mensajes)

## CONTACTO

\*Nombre

\*Email

\*Teléfono

\*Asunto

\*Mensaje

☐ He leído y acepto las [condiciones](#)

Enviar

#### 5.1.4 FOOTER

Hemos establecido un footer, con los datos de contacto y la ubicación del negocio:

## 5.2 REGISTRO

### REGISTRATE

Nombre

pablo

Apellido

beneitez

Usuario

pablo193

Contraseña

.....|

Registrarse

¿Ya tienes una cuenta? [Inicia sesión.](#)

Con las librerías AlertifyJS y Ajax de JavaScript, hemos generado una alerta personalizada para cuando nos registremos y se detecte que lo hemos hecho con un usuario que ya existe en nuestra BBDD, así como, una pequeña leyenda que nos indica que nos hemos registrado correctamente.

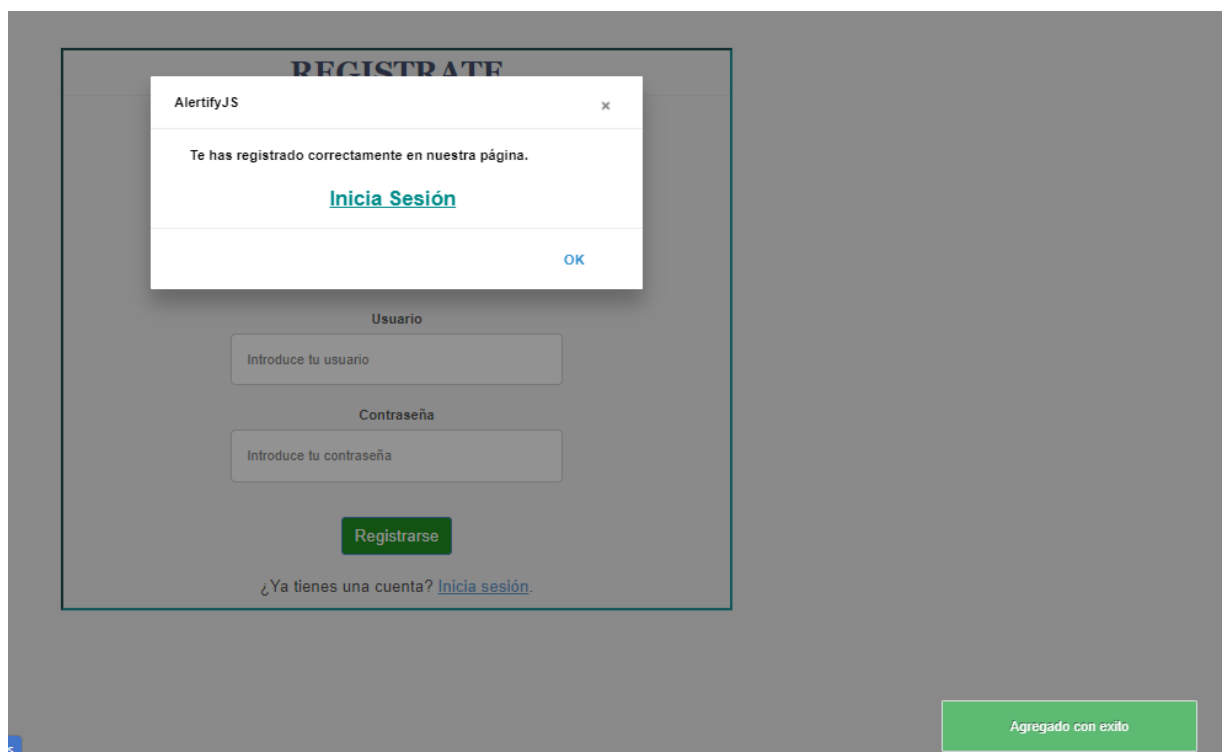
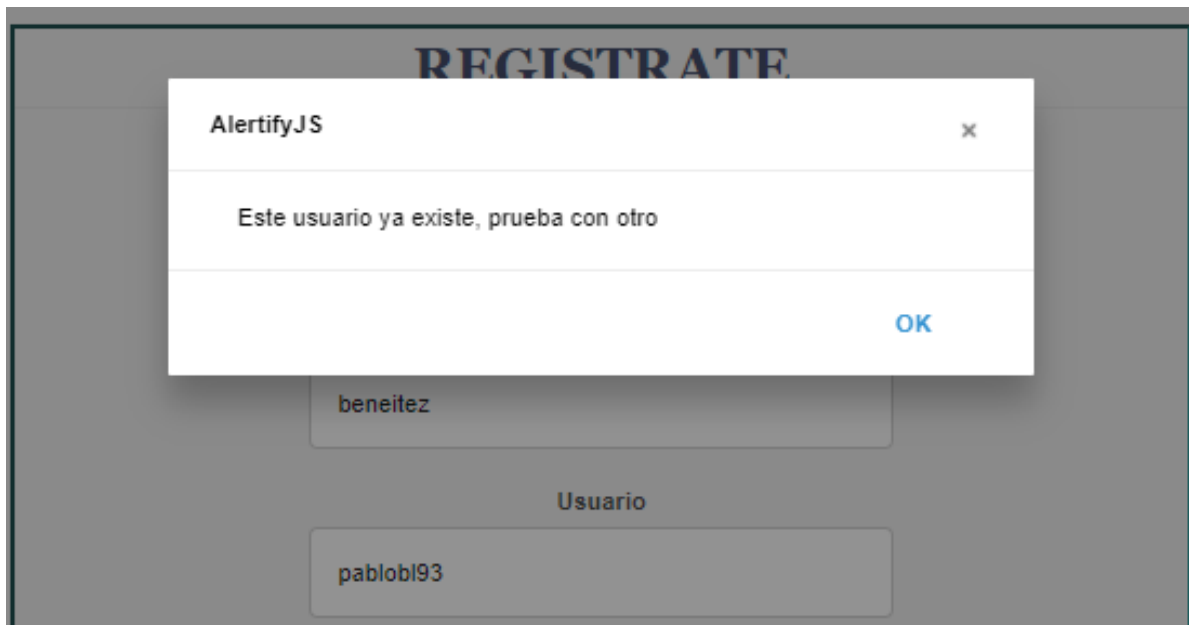
```
cadena="nombre=" + $('#nombre').val() +
"&apellido=" + $('#apellido').val() +
"&usuario=" + $('#usuario').val() +
"&password=" + $('#password').val();

$.ajax({
  type:"POST",
  url:"php/registro.php",
  data:cadena,
  success:function(r){
    if(r==2){
      alertify.alert("Este usuario ya existe, prueba con otro");
    }
    else if(r==1){
      $('#frmRegistro')[0].reset();
      alertify.success("Agregado con exito");

      alertify.alert("<b>Te has registrado correctamente en nuestra página.  
<br><br><center><a href='acceso.php#acceso_usuarios'  
style='color:darkcyan;font-size:150%;'>Inicia Sesión</a></b>");
    }else{
      alertify.error("Fallo al agregar");
    }
  }
});|
```

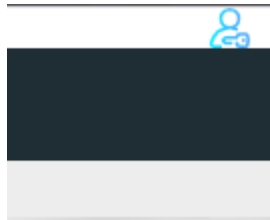


Con Ajax enviamos los datos (**data:cadena**) con los valores de nombre,apellido,usuario y password, mediante **POST**, a la url para su comprobación "**php/registro.php**" y la función success que nos devuelve 0 = fallo al agregar, si nos devuelve 2 = ya existe un usuario con ese nombre y 1 para registro con éxito.



## 5.3 INICIO DE SESIÓN

Después de registrarnos en la página, podemos iniciar sesión directamente en el cuadro de texto (imagen anterior) que nos apareció al registrarnos correctamente o en el icono en la esquina superior derecha de cualquier página de nuestro sitio.



Al situar el ratón sobre el icono, se despliega un cuadro de texto con el enlace (con las propiedades *css: hover + transition*)\* que nos lleva a 'acceso.php' donde podemos iniciar sesión.



```
.overlay {  
    position: absolute;  
    bottom: 0;  
    left: 100%;  
    background-color: darkcyan;  
    overflow: hidden;  
    width: 0;  
    height: 100%;  
    transition: .8s ease;  
}
```

```
.container:hover .overlay{  
    width: 100px;  
    left: 0;
```

```
<!-- LOGIN BUTTON -->  
<div style="float:right;">  
  <div class="container">  
      
    <div class="overlay">  
      <div class="text"><a class="login_link" href="login/acceso.php#acceso_usuarios">Iniciar Sesión</a>  
    </div>  
  </div>  
</div>  
</div>
```

Para validar los datos, hemos utilizado AJAX, con la siguiente función, el script realiza lo siguiente:

- 1- Comprueba si hemos hecho click en el input id = EntrarSistema.
- 2- Si falta algún dato, como usuario o password nos alertará de que falta este campo para rellenar.
- 3- Una vez completado ambos campos, esta función nos recoge a modo de cadena los valores usuario y password y los envía a la página login.php para allí compararlos e iniciar sesión si se corresponde correctamente con la base de datos:
- 4- Si la función success es igual a 1, es correcto el inicio y nos lleva a la página inicio.php.

```
<script type="text/javascript">
    $(document).ready(function(){
        $('#entrarSistema').click(function(){
            if($('#usuario').val()==""){
                alertify.alert("Debes agregar el usuario");
                return false;
            }else if($('#password').val()==""){
                alertify.alert("Debes agregar el password");
                return false;
            }

            cadena="usuario=" + $('#usuario').val() +
                "&password=" + $('#password').val();

            $.ajax({
                type:"POST",
                url:"php/login.php",
                data:cadena,
                success:function(r){
                    if(r==1){
                        window.location="../login/acceso_usuarios/inicio.php";
                    }else{
                        alert("El usuario/contraseña es incorrecto");
                    }
                }
            });
        });
    });
</script>
```

## 5.4 ACCESO USUARIOS

Login.php: Se envía la consulta SQL creada y si el resultado de la query con la función `mysqli_num_rows` es mayor que 0, quiere decir que encontró un usuario y si es 0, que no lo encontró.

```
<?php

session_start();
require_once "conexion.php";


$conexion=conexion();

$usuario=$_POST['usuario'];
$pass=sha1($_POST['password']);

$sql="SELECT * from usuarios where usuario='$usuario' and password='$pass'";
$result=mysqli_query($conexion,$sql);

if(mysqli_num_rows($result) > 0){
    $_SESSION['user']=$usuario;
    echo 1;
}else{
    echo 0;
}
```

### ACCESO USUARIOS



Usuario

Contraseña

[¿No tienes cuenta?, REGISTRATE](#)

[¿Eres trabajador del hotel? Inicia sesión](#)

Enviar

Una vez se realizan las comprobaciones en el backend de la página, si todo es correcto, nos aparecerá la página inicio.php, que solo podemos acceder a ella si la sesión se ha abierto.

```
<?php

session_start();

if(isset($_SESSION['user'])) {

?>

<!DOCTYPE html>
<html>
<head>
```

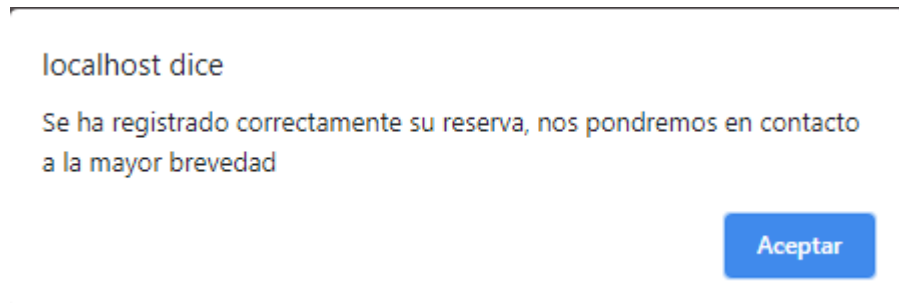


En la cuenta de cliente, existen dos opciones:

**-Realizar reserva**

The screenshot shows the 'REALIZA TU RESERVA' form. At the top, the 'Hotel Sierra de Gata' logo is visible. Below the logo, the text 'Usuario: pablobl93' is displayed. The form itself is a white box with a rounded top. Inside the box, the title 'REALIZA TU RESERVA' is centered. Below the title, there are four input fields. The first field is labeled 'Nombre' and contains the text 'PABLO'. The second field is labeled 'Apellidos' and contains the text 'BL'. The third field is labeled 'Teléfono' and contains the text '666000666'. The fourth field is labeled 'Email' and contains the text 'pablobl93@gmail.com'. The email field is highlighted with a light blue background.

Si hemos rellenado todos los campos existentes en el formulario de reserva y todo ha funcionado correctamente, aparecerá un alert que nos indica que nuestra reserva se ha registrado correctamente.



Para ello, hemos precisado de PHP, registrando en variables los valores de los inputs del formulario e insertándolo en la tabla reservas.

```
require_once "conexion.php";
$conexion=conexion();
session_start();

$usuario= ($_SESSION['user']);
$nombre=$_POST['nombre'];
$apellidos=$_POST['apellidos'];
$telefono=$_POST['telefono'];
$email=$_POST['email'];
$calle=$_POST['calle'];
$calle_numero=$_POST['calle_numero'];
$ciudad=$_POST['ciudad'];
$codigo_postal=$_POST['codigo_postal'];
$pais=$_POST['pais'];
$fecha_entrada=$_POST['fecha_entrada'];
$fecha_salida=$_POST['fecha_salida'];
$personas=$_POST['personas'];
$habitaciones=$_POST['habitaciones'];
$servicios=$_POST['servicios'];

$sql = "INSERT INTO reservas (usuario,nombre,apellidos,telefono,email,calle,calle_numero,
ciudad,codigo_postal,pais,fecha_entrada,fecha_salida,personas,habitaciones,servicios)
VALUES ('$usuario','$nombre','$apellidos','$telefono','$email','$calle','$calle_numero', '$ciudad','$codigo_posta

if (mysqli_query($conexion, $sql)) {
    echo'<script type="text/javascript">
    alert("Se ha registrado correctamente su reserva, nos pondremos en contacto a la mayor brevedad");
    window.location.href="inicio.php";
    </script>';
}
```

## -Consultar reserva

Del mismo modo y con PHP podemos consultar las reservas que tenemos:



The screenshot shows the Hotel Sierra de Gata logo at the top. Below it, the user is identified as 'Usuario: pablobl93'. The message 'Tus reservas son las siguientes:' is displayed above a table of reservations. The table has columns for ID\_RESERVA, USUARIO, NOMBRE, APELLIDOS, TELEFONO, EMAIL, FECHA ENTRADA, FECHA SALIDA, PERSONAS, HABITACION, and SERVICIOS. Two reservations are listed: ID 22 for 'Probando' and ID 23 for 'PABLO'. Below the table are two buttons: 'Volver a mi cuenta' and 'Salir del sistema'.

ID_RESERVA	USUARIO	NOMBRE	APELLIDOS	TELEFONO	EMAIL	FECHA ENTRADA	FECHA SALIDA	PERSONAS	HABITACION	SERVICIOS
22	pablobl93	Probando	COPIA SEGURIDAD	690645	pablobl93@gmail.com	2021-05-19	2021-05-20	2	Suite	Rutas
23	pablobl93	PABLO	BL	666000666	pablobl93@gmail.com	2021-05-20	2021-05-21	2	Suite	Paintball

Resultado de tabla reservas en phpmyadmin:

+ Opciones

	ID	usuario	nombre	apellidos	telefono	email	calle	calle_numero	ciudad	codigo_postal	pais	fecha_entrada	fecha_salida	p
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	22	pablobl93	Probando	COPIA SEGURIDAD	690645	pablobl93@gmail.com	AVDA AMERICA	24	León	24402	España	2021-05-19	2021-05-20	
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	23	pablobl93	PABLO	BL	666000666	pablobl93@gmail.com	avda america	55	Ponferrada	24402	España	2021-05-20	2021-05-21	

☐ Seleccionar todo Para los elementos que están marcados: ☐ Editar ☐ Copiar ☐ Borrar ☐ Exportar

En el caso de que no existieran reservas, hemos utilizado jQuery para hacer más pequeño el div principal donde se posiciona el texto.



The screenshot shows the Hotel Sierra de Gata logo at the top. Below it, the user is identified as 'Usuario: pablobl93'. The message 'No existen reservas asociadas a 'pablobl93'' is displayed. Below the message are two buttons: 'Volver a mi cuenta' and 'Salir del sistema'.

Se ha realizado con un elseif dentro de una sentencia condicional, en el caso de que número de columnas(num\_rows) del resultado sea igual a 0, realizamos el script donde se intercambia el valor "height" a 15 em.

```
<?php

}

elseif ($resultado->num_rows == 0)
{
    print "<h2><center>No existen reservas asociadas a '$usuario'</center></h2>";
    echo'<script type="text/javascript">
    $(document).ready(function() {
    $(".welcome-div").css({ "width":"99%", "height":"15em" });
    })</script>';
}
else
{
    $resultado=$bd->query("Select * from reservas where usuario='$usuario'");
    $datos=$resultado->fetch_array();

    print ("<h1 style=color:#898989;><center>Tus reservas son las siguientes: <br><br></cen
    print "<table border='2' class='table_result' >";
    print "<tr>";
    print "<th>ID_RESERVA";
    print " <th> USUARIO";
    print "<th>NOMBRE";
    print "<th>APELLIDOS";
    print "<th>TELEFONO";
```

Además, ambos botones, volver a mi cuenta y salir del sistema, referencian a las páginas inicio.php y salir.php respectivamente.

## 5.5 CUENTA DE EMPLEADO

### ACCESO EMPLEADOS



Usuario

Contraseña



Con PHP hemos obtenido el nombre del empleado para mostrarlo en el menú inicio:



```
<?php
    $resultado=$bd->query("Select * from empleados where usuario='$usuario'");
    $datos=$resultado->fetch_array();
    $nom=$datos[3];
    $apell=$datos[4];

?>
    <span class="welcome-span">Bienvenido <?php echo $nom ; echo " " . $apell; ?></span>
```

CSS aplicado ("welcome-span") ("welcome-div"):

```
.welcome-span {
    background-color: #EED984;
    font-size: 220%;
    color: #615F58;
    font-family: 'Rokkitt', Georgia, Times New Roman, serif;
    font-weight: bold;
}
```

```
.welcome-div {
    background-color: #DADEDE;
    width: 50%;
    height: 15em;
    margin-left: auto;
    margin-right: auto;
}
```

En la cuenta de empleado, existen dos opciones:

### -Consultar horario de trabajo



The screenshot shows the Hotel Sierra de Gata logo at the top. Below it, the user is identified as 'Usuario: V323456'. The text 'Tus horarios son los siguientes:' is displayed. A table shows the employee's schedule:

ID_EMPLEADO	USUARIO	NOMBRE	APELLIDOS	HORARIO_ENTRADA	HORARIO_SALIDA
3	V323456	Luis	Gómez	09:00:00	17:00:00

At the bottom, there are two buttons: 'Volver a mi cuenta' and 'Salir del sistema'.

```
else
{
$resultado=$bd->query("Select * from empleados where usuario='$usuario'");
$datos=$resultado->fetch_array();
/*SCRIPT QUE CAMBIA EL TAMAÑO DEL DIV*/
echo'<script type="text/javascript">
$(document).ready(function() {
$(".welcome-div").css({ "width":"99%", "height":"25em" });
})</script>';
print("<h1 style=color:#898989;><center>Tus horarios son los siguientes: <br><br></center>");
print("<table border='2' class='table_result' >");
print "<tr>";
print "<th>ID_EMPLEADO";
print " <th> USUARIO";
print "<th>NOMBRE";
print "<th>APELLIDOS";
print "<th>HORARIO_ENTRADA";
print "<th>HORARIO_SALIDA";
print "</th>";
print "</tr>";
print "<td>".$datos[0]."</td>";
print "<td>".$datos[1]."</td>";
print "<td>".$datos[3]."</td>";
print "<td>".$datos[4]."</td>";
print "<td>".$datos[5]."</td>";
print "<td>".$datos[6]."</td>";
print "</table>";
}
```

## -Consultar condiciones laborales



Usuario: **V323456**

### CONDICIONES LABORALES:

ID_EMPLEADO	USUARIO	NOMBRE	APELLIDOS	PUESTO	CONTRATO	SUELDO
3	V323456	Luis	Gómez	Técnico Contable	Indefinido	1450

[Volver a mi cuenta](#)

[Salir del sistema](#)

El mismo proceso se ha realizado para obtener las condiciones laborales almacenadas en la tabla empleados

```
print ("<h1 style=color:#898989;><center>CONDICIONES LABORALES: <br><
print "<table border='2' class='table_result' >";
    print "<tr>";
    print "<th>ID_EMPLEADO";
    print "<th>USUARIO";
    print "<th>NOMBRE";
    print "<th>APELLIDOS";
    print "<th>PUESTO";
    print "<th>CONTRATO";
    print "<th>SUELDO";
    print "</th>";
    print "</tr>";
    /*RESULTADOS QUERY*/
    print "<td>".$datos[0]."</td>";
    print "<td>".$datos[1]."</td>";
    print "<td>".$datos[3]."</td>";
    print "<td>".$datos[4]."</td>";
    print "<td class='puesto'>".$datos[7]."</td>";
    print "<td class='contrato'>".$datos[8]."</td>";
    print "<td class='sueldo'>".$datos[9]."</td>";
print "</table>";
```

## 6. SERVIDOR WEB EN RASPBERRY

### 6.1 ESPECIFICACIONES

La [Raspberry Pi 3 Model +](#) que hemos adquirido, tiene las siguientes especificaciones:

- CPU + GPU: Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- RAM: 1GB LPDDR2 SDRAM
- Wi-Fi + Bluetooth: 2.4GHz y 5GHz IEEE 802.11.b/g/n/ac, Bluetooth 4.2, BLE
- Ethernet: Gigabit Ethernet sobre USB 2.0 (300 Mbps)
- GPIO de 40 pines
- HDMI
- puertos USB 2.0
- Puerto CSI y DSI para conectar una cámara y una pantalla táctil
- Salida de audio estéreo y vídeo compuesto
- Micro-SD
- Power-over-Ethernet (PoE)

Imprescindible adquirir también los [Accesorios Raspberry:](#)

Contenido del paquete:

- 1x Aukru 5V 3A cargador
- 1x caja transparente
- 3x disipador de calor



## 6.2 INSTALACIÓN DEL HARDWARE

Apache es un servidor web de código abierto, multiplataforma y gratuito.

Uno de los más utilizados en todo el mundo y que se ha instalado en la Raspberry para poder alojar en la misma la página web.

Al solo tener una Raspberry física para hacer simulaciones, hemos instalado Apache / Servidor Web en ella.

A continuación, una foto de la Raspberry real:



Para poder conectar este dispositivo, necesitamos la caja portable, la fuente de alimentación y los disipadores térmicos, que, anteriormente, compramos en Amazon

Ya con la carcasa y disipadores instalados.



## 6.3 PASOS EN LA INSTALACIÓN

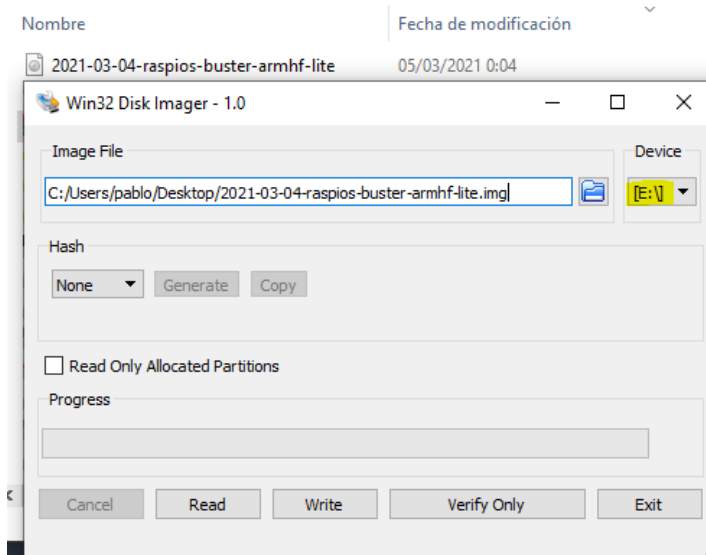
Antes de conectarla a la red, debemos instalar el sistema operativo elegido en la [Tarjeta SD 32 GB](#): previamente adquirida.

Elegimos el sistema operativo propicio para este tipo de mini-ordenadores, **Raspbian**, basado en la distribución de Linux, Debian.

Nos descargamos de la web oficial la imagen con el sistema operativo "[Raspberry Pi Os Lite](#)", que nos ocupa 442 MB.

Para poder montar una imagen de sistema operativo en la tarjeta SD, debemos conectar la misma al PC y con un programa especializado llamado [WIN32 Disk Imager](#), que es una herramienta de Windows para escribir imágenes de sistemas operativo en memorias USB o tarjetas SD.

Seleccionamos la ruta donde tenemos la imagen en C: y establecemos la instalación en E:, que es donde está conectada la tarjeta:




Para comprobar que el sistema operativo se ha instalado correctamente, debemos conectar la Raspberry a la toma de luz y acoplarle el cable HDMI para poder transmitir la imagen a una pantalla:



Iniciamos el dispositivo, con las credenciales, usuario: **pi** y contraseña: **Raspberry**, después cambiaremos esta contraseña puesto que todas las Raspberrys vienen con las mismas credenciales de acceso, por lo tanto, generaremos una contraseña lo suficientemente fuerte para aumentar la seguridad del dispositivo.

Comprobaremos la password elegida, en la web de [Kaspersky](https://www.kaspersky.com), en su apartado de comprobación de seguridad de claves:



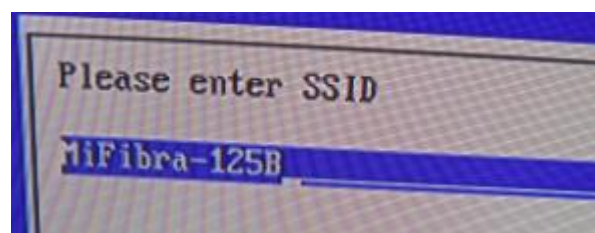
 **¡Buena contraseña!**

- Tu contraseña es resistente al pirateo.
- Tu contraseña no aparece en ninguna base de datos de contraseñas filtradas.

Tu contraseña puede ser descifrada con un ordenador común en...

4 años

Para poder configurar nuestra Raspberry, usaremos el comando *raspi-config* para implantar la conexión Wifi con la red local de casa (Mi Fibra-125B), escribimos la SSID de nuestra Wifi y la password para conectarnos:





red cableada

### Servicios

- Internet **disponible**
- teléfono **disponible**  
No. [redacted]
- TV **disponible**

superior - Safe  
superior - Ope  
10.00 y superi

 ¿Sabías?

Puedes acced  
contenido en l  
disco duro ext  
conectado a la  
compartirlo de  
ordenador.

Wi-Fi **activado**  
red Wi-Fi

<< LAPTO... Galax... raspb... >>

**raspberrypi**

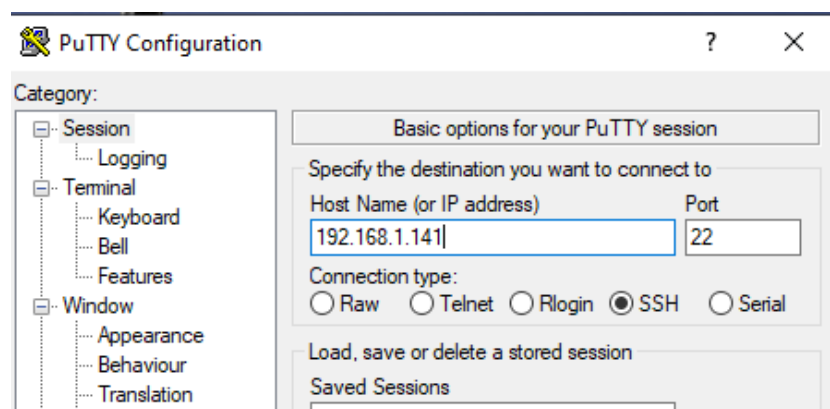


## 6.4 ACCESO DESDE CLIENTE SSH

Nos conectaremos al futuro servidor web desde un cliente **SSH**(Secure SHell )como Putty, que nos permite “manejarlo” desde nuestro PC y es mucho más cómodo que conectando a la propia Raspberry un teclado y un ratón.

Por motivos de seguridad, en las últimas versiones de Raspberry, la conexión por SSH está deshabilitada, debemos activarla con el mismo comando anterior `raspi-config` y en servicio SSH marcamos <Enable>.

Para acceder, primero debemos saber la IP de nuestro dispositivo, con el comando **if-config**, la IP que nuestra red ha asignado al dispositivo es 192.168.1.141.



## 6.5 INSTALACIÓN SERVIDOR WEB

Los dos primeros comandos que debemos usar son: `apt-get update` (actualizar lista de paquetes de repositorios de Raspbian) y `apt-get upgrade` (actualizar todos los paquetes que tenemos instalados)

### Instalación Apache y PHP 5

Se instalan los paquetes necesarios para crear el servidor web.

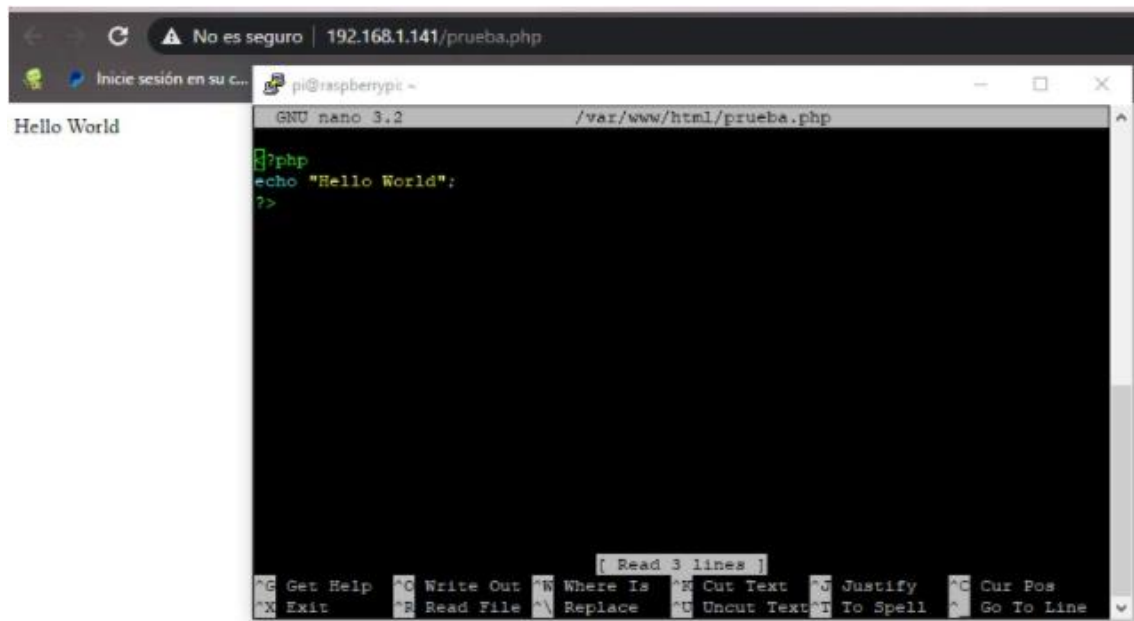
Instalamos los paquetes necesarios para tener Apache (`sudo apt-get install apache2`) y PHP (para que nuestro “hosting” pueda servir páginas dinámicas).

Primeramente, vamos a instalar los paquetes necesarios para tener Apache en nuestro servidor web casero. También PHP para que nuestro hosting gratuito pueda servir páginas dinámicas y mysql que es un motor de bases de datos que funciona muy bien con PHP.

```
root@raspberrypi:/home/pi# apt install php libapache2-mod-php php-mysql
```



Para verificar si hemos instalado correctamente PHP en nuestro servidor, haremos una pequeña prueba, dentro del directorio **/var/www/html/**, situaremos un archivo php para comprobar si se ejecuta esta página, y, para comprobarlo en un navegador propio de la red, en este caso desde el mismo PC escribiremos la dirección IP de la Raspberry 192.168.1.141 seguido del archivo que hemos creado (prueba.php):



## Instalación de MySQL

Con los siguientes comandos instalamos el motor de base de datos MySQL:

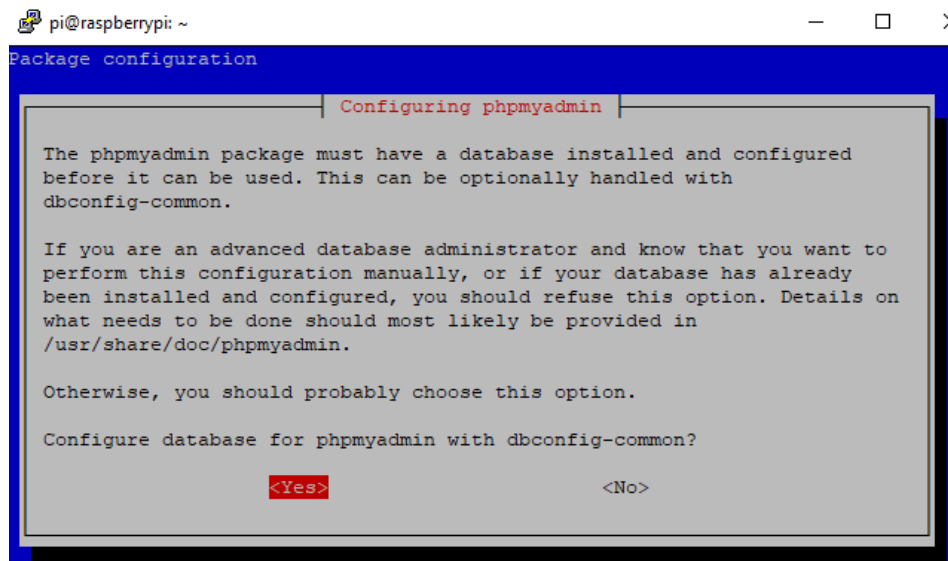
```
Processing triggers for libc-bin (2.28-10+rpi1) ...
root@raspberrypi:/home/pi# apt-get install mariadb-client-10.0
```

```
Processing triggers for libc-bin (2.28-10+rpi1) ...
root@raspberrypi:/home/pi# apt-get install mariadb-server-10.0
```

Para facilitarnos la gestión de la base de datos y su manejo, vamos a instalar PhpMyAdmin con el siguiente comando:

```
Processing triggers for libc-bin (2.28-10+rpi1) ...
root@raspberrypi:/home/pi# apt-get install phpmyadmin
```

En la instalación de MySQL nos aparecen ventanas emergentes a modo de configuración, entre ellas, establecer la contraseña para root:



Para terminar, con el comando `systemctl status` (nombre de servicio) comprobamos cual es el estado de nuestro servicio instalado:

```
pi@raspberrypi:~ $ systemctl status mysql.service
● mysql.service - LSB: Start and stop the mysql database server daemon
   Loaded: loaded (/etc/init.d/mysql; generated)
   Active: active (running) since Thu 2021-05-13 14:47:04 BST; 7min ago
     Docs: man:systemd-sysv-generator(8)
    Tasks: 26 (limit: 2062)
   CGroup: /system.slice/mysql.service
           └─16529 /bin/bash /usr/bin/mysqld_safe
             └─16674 /usr/sbin/mysqld --basedir=/usr --datadir=/var/lib/mysql --pl
               └─16675 logger -t mysqld -p daemon error

May 13 14:47:07 raspberrypi /etc/mysql/debian-start[16726]: Processing databases
May 13 14:47:07 raspberrypi /etc/mysql/debian-start[16726]: information_schema
May 13 14:47:07 raspberrypi /etc/mysql/debian-start[16726]: mysql
May 13 14:47:07 raspberrypi /etc/mysql/debian-start[16726]: performance_schema
May 13 14:47:07 raspberrypi /etc/mysql/debian-start[16726]: Phase 5/6: Checking
May 13 14:47:07 raspberrypi /etc/mysql/debian-start[16726]: Processing databases
May 13 14:47:07 raspberrypi /etc/mysql/debian-start[16726]: information_schema
May 13 14:47:07 raspberrypi /etc/mysql/debian-start[16726]: performance_schema
May 13 14:47:07 raspberrypi /etc/mysql/debian-start[16726]: Phase 6/6: Running
May 13 14:47:07 raspberrypi /etc/mysql/debian-start[16726]: OK
lines 1-20/20 (END)
```

Para poder ver phmyadmin de forma gráfica, debemos editar el archivo de configuración del servidor Web Apache

```
pi@raspberrypi:~ $ sudo nano /etc/apache2/apache2.conf
```

```
pi@raspberrypi: ~
GNU nano 3.2 /etc/apache2/apache2.conf Modified
#
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" $
LogFormat "%h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" combin$
LogFormat "%h %l %u %t \"%r\" %>s %O" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

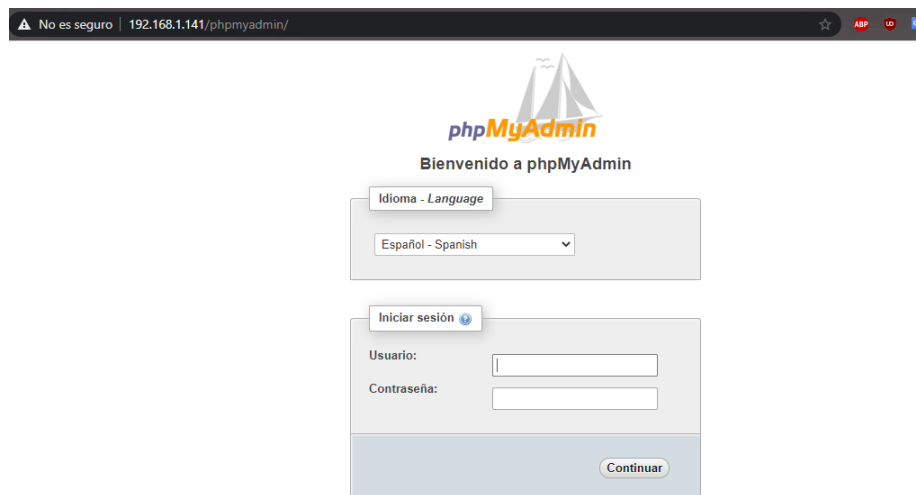
# Include of directories ignores editors' and dpkg's backup files,
# see README.Debian for details.

# Include generic snippets of statements
IncludeOptional conf-enabled/*.conf

# Include the virtual host configurations:
IncludeOptional sites-enabled/*.conf

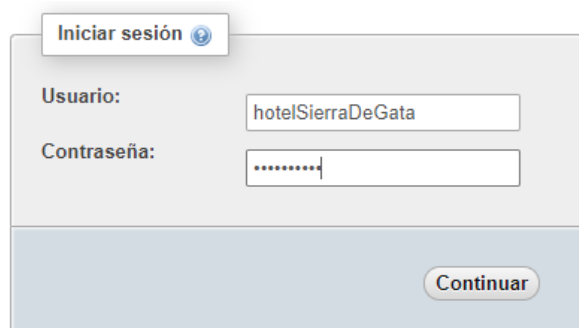
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
include /etc/phpmyadmin/apache.conf
```

Incluimos la última línea en el archivo de configuración y reiniciamos el servicio apache con *sudo service apache2 restart*:



Vamos a agregar por la línea de comandos un usuario llamado hotelSierraDeGata con todos los permisos, ya que será el usuario del trabajador que se encargue de la parte informática de la empresa:

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'hotelSierraDeGata'@'localhost' IDENTIFIED BY '1234567890';
```



```

MariaDB [(none)]> select user from mysql.user;
+-----+
| user |
+-----+
| hotelSierraDeGata |
| pablo |
| phpmyadmin |
| root |
+-----+
4 rows in set (0.001 sec)

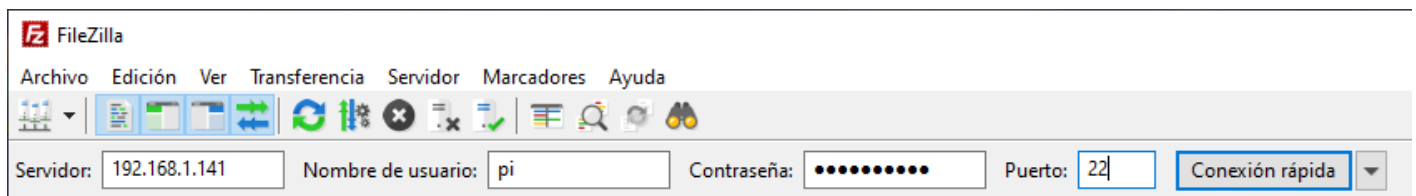
MariaDB [(none)]>

```

## 6.6 ACCESO POR FTP AL SERVIDOR

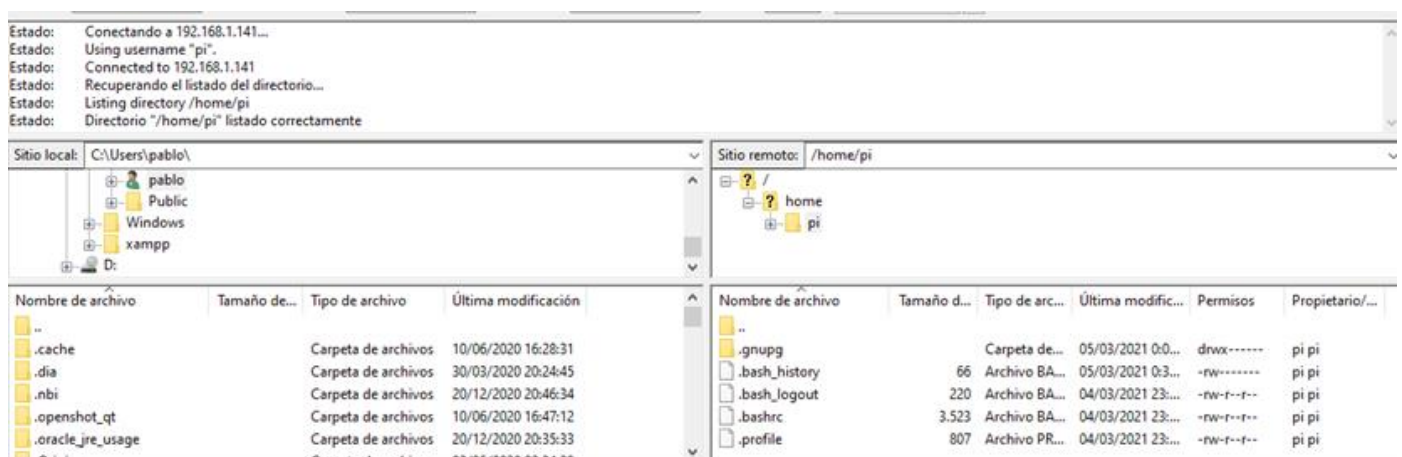
Para poder subir nuestra web que ya hemos hecho anteriormente con Netbeans, debemos instalar un cliente FTP (FileZilla) para alojar en el servidor todas las carpetas de nuestro sitio.

Una vez iniciado el FTP, rellenaremos los siguientes campos para conectarnos a nuestro server:

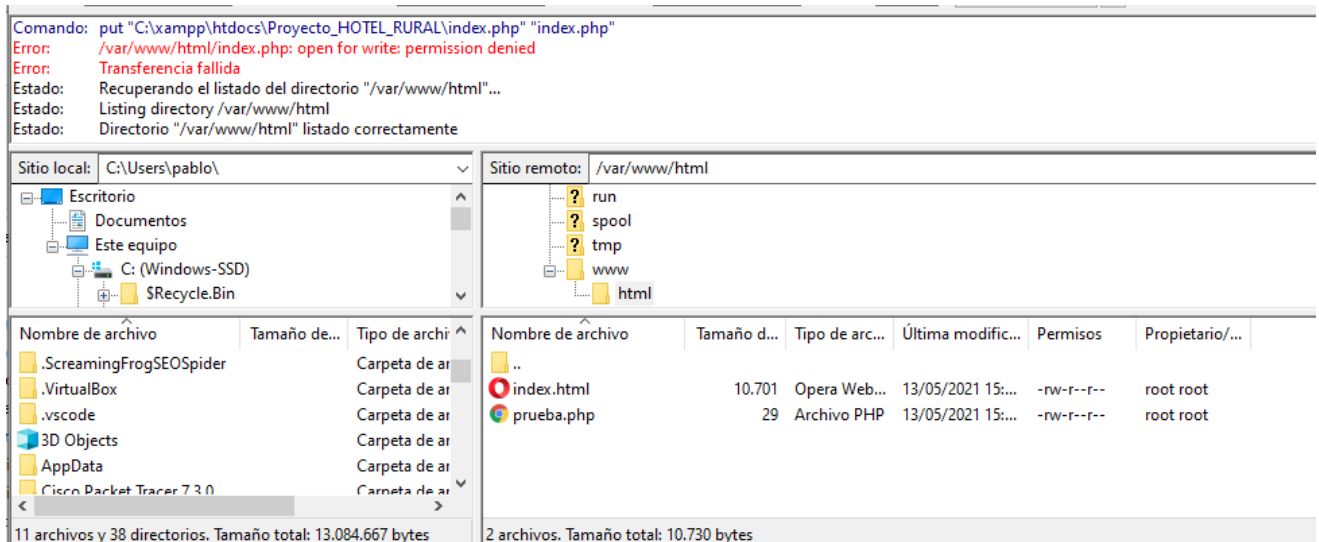


- Servidor: La IP de nuestro servidor 192.168.1.141
- Usuario: pi
- Contraseña: La establecida para el usuario pi
- Puerto: Elegimos el puerto 22, igual que con el cliente SSH-Putty, para así realizar conexiones seguras.

Si hemos realizado correctamente la configuración de acceso nos dejará listar los archivos que tenemos en el servidor:



Detectamos un problema al intentar arrastrar la carpeta de nuestra web al servidor, y es que, nos aparece un error de operación no permitida, debido a que hemos accedido con el usuario pi y nuestras acciones sobre los directorios están limitadas, volveremos entonces a Putty para comprobar los permisos del usuario.



Hacemos un ls al directorio donde queremos alojar nuestra web, se observa que tiene permisos de escritura, lectura y ejecución para el dueño del directorio (www-data; que es el usuario que utiliza apache), para grupo (www-data) tiene permisos de lectura y ejecución y para otros usuarios permisos de ejecución:

```
root@raspberrypi:/home/pi# ls -l /var/www
total 4
drwxrwx--- 6 www-data www-data 4096 May 16 12:25 html
root@raspberrypi:/home/pi#
```

Añadimos al grupo www-data el usuario pi, con el siguiente comando:

```
pi@raspberrypi:~$ sudo usermod pi -G www-data
```

Para comprobarlo, utilizamos el comando `id -gn pi`

```
pi@raspberrypi:~$ id -gn pi
www-data
```

Abrimos nuevamente el cliente FTP para comprobar que con el usuario pi tenemos todos los permisos (aunque el fundamental es el permiso de escritura para poder “mover” nuestra web a la Raspberry)

Al añadir el usuario pi al grupo www-data, nos ha dejado sin problemas listar y añadir la carpeta de nuestro proyecto “HotelSierraDeGata” al directorio /var/www/html

Servidor: sftp://192.168.1.141 Nombre de usuario: pi Contraseña: ..... Puerto: Conexión rápida

Estado: Recuperando el listado del directorio "/var/www/html"...

Estado: Listing directory /var/www/html

Estado: Directorio "/var/www/html" listado correctamente

Estado: Recuperando el listado del directorio "/var/www/html/hotelSierraDeGata"...

Estado: Listing directory /var/www/html/hotelSierraDeGata

Estado: Directorio "/var/www/html/hotelSierraDeGata" listado correctamente

Sitio local: C:\Users\pablo\

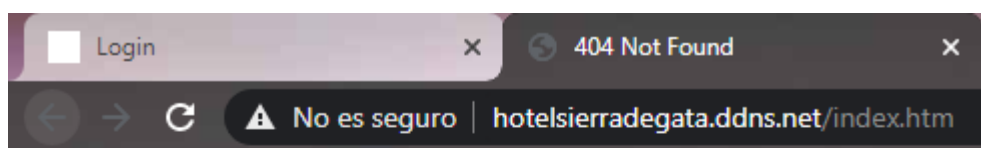
Sitio remoto: /var/www/html/hotelSierraDeGata

Nombre de archivo	Tamaño de...	Tipo de archivo	Nombre de archivo	Tamaño d...	Tipo de arc...	Última modific...	Permisos	Propietario/...
..		Carpeta de archivos	..					
.cache		Carpeta de archivos	images		Carpeta de...	21/05/2021 16:...	drwxr-xr-x	pi www-data
.dia		Carpeta de archivos	login		Carpeta de...	21/05/2021 16:...	drwxr-xr-x	pi www-data
.nbi		Carpeta de archivos	nbproject		Carpeta de...	21/05/2021 16:...	drwxr-xr-x	pi www-data
.openshot_qt		Carpeta de archivos	contacto.php	16.887	Archivo PHP	21/05/2021 16:...	-rw-r--r--	pi www-data
.oracle_jre_usage		Carpeta de archivos	entorno.php	17.447	Archivo PHP	21/05/2021 16:...	-rw-r--r--	pi www-data
.Origin		Carpeta de archivos	galeria.php	18.568	Archivo PHP	21/05/2021 16:...	-rw-r--r--	pi www-data
.QtWebEngineProcess		Carpeta de archivos	galeria2.php	18.884	Archivo PHP	21/05/2021 16:...	-rw-r--r--	pi www-data
.ScreamingFrogSEOSpider		Carpeta de archivos	habitaciones.php	17.757	Archivo PHP	21/05/2021 16:...	-rw-r--r--	pi www-data
			historia.php	17.389	Archivo PHP	21/05/2021 16:...	-rw-r--r--	pi www-data

11 archivos y 38 directorios. Tamaño total: 12.326.907 bytes

10 archivos y 3 directorios. Tamaño total: 176.968 bytes

Para visualizar la página web desde cualquier dispositivo de nuestra red local, debemos otorgar permisos al resto de usuarios:



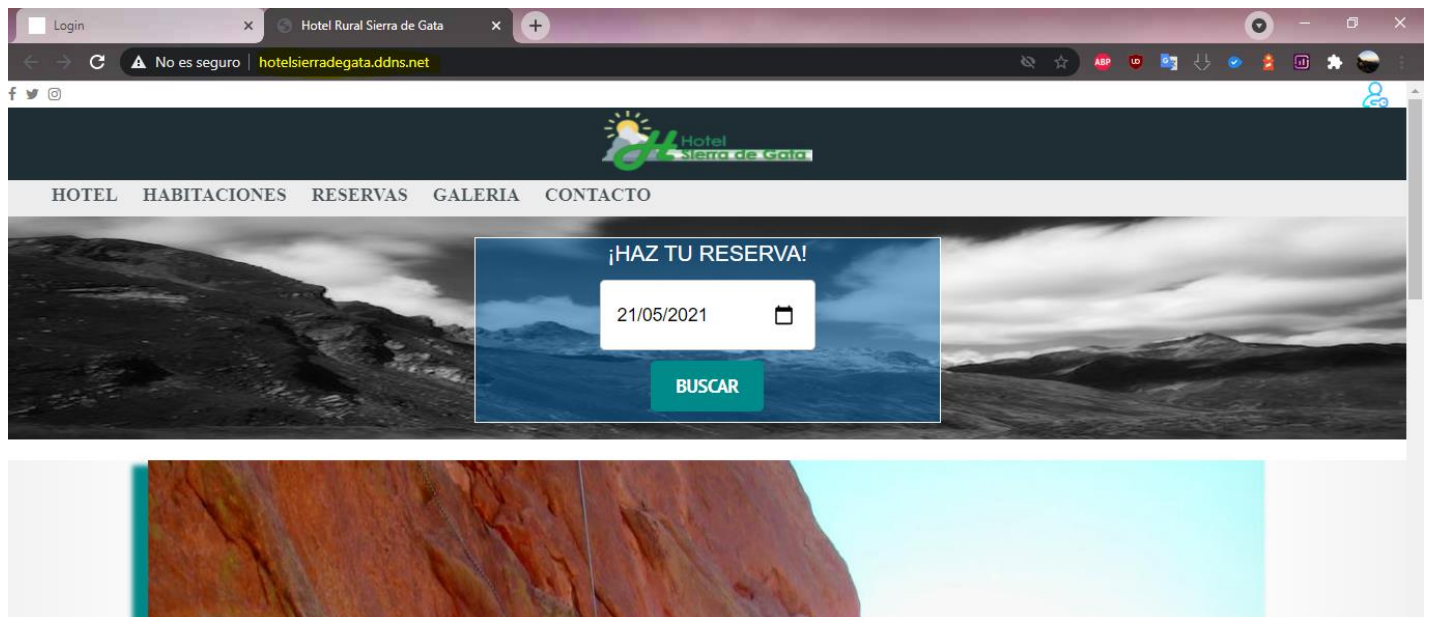
## Not Found

The requested URL was not found on this server.

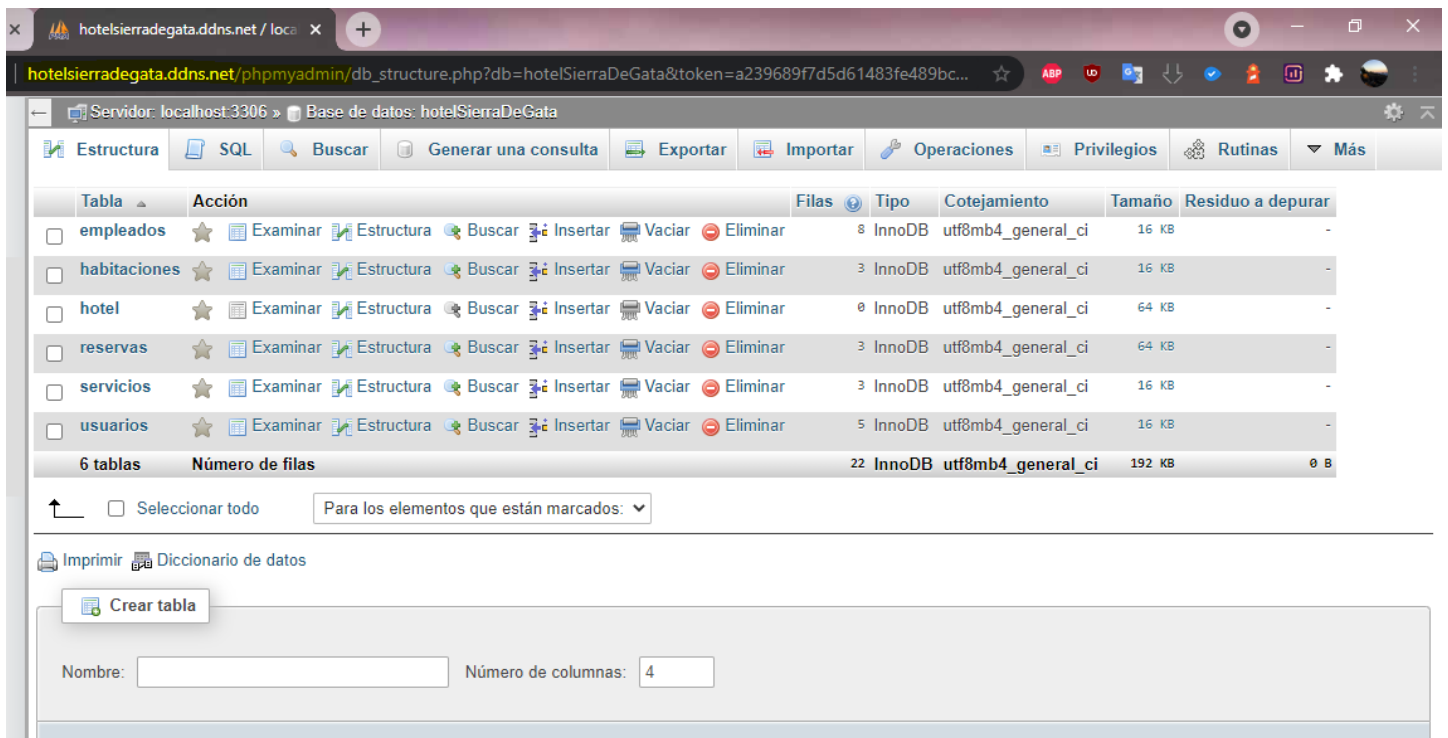
Apache/2.4.38 (Raspbian) Server at hotelsierradegata.ddns.net Port 80

```
root@raspberrypi:/home/pi# chmod 777 /var/www
```

Después de aplicar los permisos correspondientes, ya podemos acceder desde nuestro PC a la página web alojada en la Raspberry.



También podremos acceder a PhpMyAdmin y ver nuestra base de datos desde: <http://hotelsierradegata.ddns.net/phpmyadmin/>



El siguiente paso será, mediante DNS dinámico, conseguir que se pueda acceder a nuestra web desde una red exterior.



## 6.7 DNS DINÁMICO CON NO-IP

Para poder acceder a nuestro sitio desde el exterior de nuestra red local, haremos uso de DNS dinámico.

### DNS DINÁMICO:

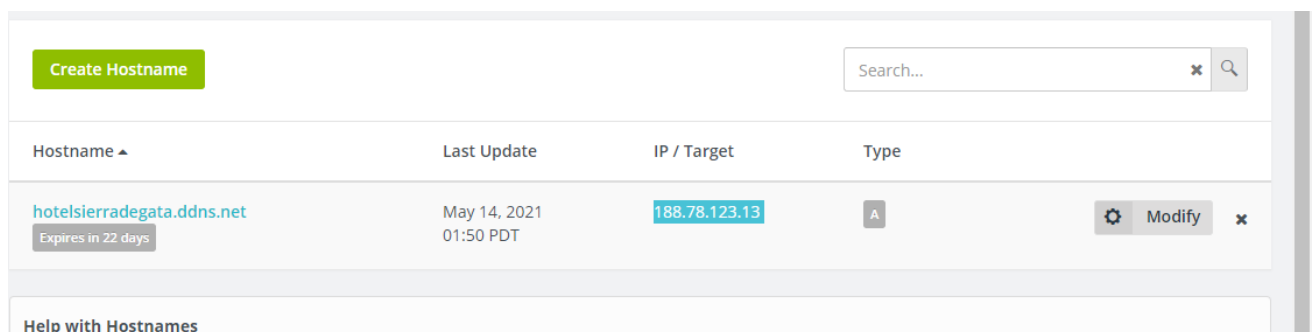
Su ventaja principal es que nos permite acceder a nuestro servidor web con una dirección fácil de recordad sin necesidad de memorizar la IP pública.

En nuestro caso, como en el de la mayoría de clientes domésticos, contamos con una IP dinámica, aunque no es problema, ya que usando este servicio, se sincronizará automáticamente con la IP dinámica que tengamos en ese momento, siempre y cuando lo configuremos en nuestro router o con la instalación del cliente No-IP.

### No-IP

Servicio gratuito donde podemos crear dominios para redireccionar conexiones, asociará nuestra IP pública con la IP de nuestro servidor web (192.168.1.141) y la registrará con un nombre de dominio que consideremos.

Crearemos una cuenta en el sitio y asociaremos el host creado a nuestra dirección IP pública



The screenshot shows the No-IP management interface. At the top left is a green 'Create Hostname' button. To its right is a search bar with the placeholder text 'Search...'. Below these is a table with the following columns: 'Hostname', 'Last Update', 'IP / Target', and 'Type'. A single row is visible with the hostname 'hotelsierradegata.ddns.net', a last update of 'May 14, 2021 01:50 PDT', and a target IP of '188.78.123.13'. The type is 'A'. To the right of the row are 'Modify' and 'x' buttons. A small badge next to the hostname says 'Expires in 22 days'. At the bottom left of the interface is a link that says 'Help with Hostnames'.

Hostname ▲	Last Update	IP / Target	Type
hotelsierradegata.ddns.net Expires in 22 days	May 14, 2021 01:50 PDT	188.78.123.13	A

Pero para que le dominio apunte a nuestra Raspberry y por tanto, al servidor web, aplicaremos unos cambios en el router:

- Configurar DynDNS (DDNS)
- Abrir puerto 80 (HTTP)



## Configuración DDNS:

Debemos indicarle al Router si contamos con algún servicio DDNS y, si es así, registrarlo:

Utilizar DynDNS puede ser útil si alojas un sitio web, servidor ftp o cualquier tipo de servidor en la LAN. Puedes encontrar fácilmente un nombre como thisiscool.dyndns.org

configuración del Servidor de Nombres Dinámico (DDNS)

servicio	nombre de host completo	nombre de usuario email	contraseña	última actualización	
dyndns	<input type="text"/>	<input type="text"/>	<input type="password"/>		guardar
NoIP	hotelsierradegata.ddns.net	proyectopablo2021@gmail.com	*****	14/05/21 10:54:46	borrar

- Servicio: NoIP
- Nombre del host: hotelsierradegata.ddns.net
- Usuario: email con el que nos registramos en el servicio NoIP
- Contraseña: email con la que accedemos

## Abrir puerto 80 – HTTP:

# Livebox Fibra

[mi red local](#) [Wi-Fi](#) [mis archivos](#) [mi teléfono](#) [información y diagnóstico](#) [configuración avanzada](#)

e-mail  
servidor de impresión

Estas normas son necesarias para autorizar una conexión remota desde Internet que llegue a un dispositivo específico de tu red LAN. También puedes definir los puertos(s) que utilizará esta comunicación.

Para crear la regla NAT debes introducir la IPv4 asignada a tu dispositivo en la LAN. Para saber cuál es puedes consultar el listado de IPs asignadas en la pestaña "DHCP" de esta página.

 Atención: Asegúrate de que no has filtrado estos puertos en el firewall.

Personalizar reglas

estado	aplicación / servicio	puerto interno	puerto externo	protocolo	IPv4 del dispositivo	
	FTP Server	21	21	TCP		añadir
	Web Server (HTTP)	80	80	both	192.168.1.141	delete

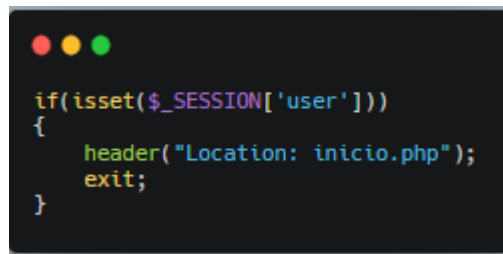
## 7. SEGURIDAD

A pesar de establecer contraseñas seguras, sería recomendable establecer más mecanismos de seguridad que se explican a continuación:

### 7.1 FORMULARIOS

En todos los formularios se ha insertado la recogida de datos del campo contraseña como **SHA1** (algoritmo de cifrado).

Además, hay páginas de la web que únicamente se puede acceder a ellas estando con la sesión iniciada (variable **\$\_SESSION**), si se intenta acceder a estas páginas sin realizar el login, surgirá un error de acceso.



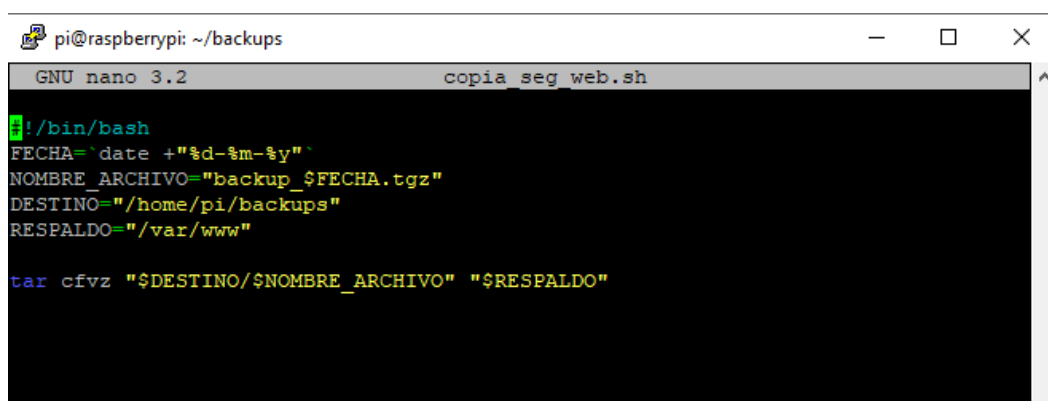
```
if(isset($_SESSION['user']))
{
    header("Location: inicio.php");
    exit;
}
```

### SEGURIDAD EN EL ACCESO A BBDD

Únicamente tienen permisos de acceso a MySQL los usuarios root y hotelSierraDeGata, a los que, obviamente hemos cambiado la contraseña de acceso.

### 7.2 COPIAS DE SEGURIDAD

Se ha programado un script que nos realiza una copia de seguridad del directorio **/var/www** y lo empaqueta con **tar**, que se utiliza para comprimir una colección de archivos y carpetas, lo guarda en la carpeta backups de pi y le añade la fecha del día en el que se realiza la copia:



```
pi@raspberrypi: ~/backups
GNU nano 3.2 copia_seg_web.sh
#!/bin/bash
FECHA=`date +%d-%m-%y`
NOMBRE_ARCHIVO="backup_`FECHA`.tgz"
DESTINO="/home/pi/backups"
RESPALDO="/var/www"

tar cfvz "$DESTINO/$NOMBRE_ARCHIVO" "$RESPALDO"
```

Lo primero que haremos será dar permisos de ejecución al propietario para poder ejecutar el script:

```
-rw-r--r-- 1 pi pi 168 May 16 12:45 copia_seg_web.sh
```

con `chmod 700` le otorgamos a pi los permisos de lectura, escritura y ejecución del archivo:

```
pi@raspberrypi:~/backups $ chmod 700 copia_seg_web.sh
pi@raspberrypi:~/backups $ ls -l
total 4
-rwx----- 1 pi pi 168 May 16 12:45 copia_seg_web.sh
```

### Automatizar copias de seguridad con Crontab:

Crontab: Es un archivo que posee una lista con todos los scripts o acciones a ejecutar.

```
pi@raspberrypi:~/backups $ crontab -e
```

Añadimos la siguiente línea al archivo porque queremos probar el script a las 16:34 del día 16. (después se estableció que se hiciese todos los días la copia del script, a la misma hora | `34 16 * * *` | )

```
)
# m h dom mon dow    command
34 16 16 * * /home/pi/backups/copia_seg_web.sh
```

Hacemos la comprobación, a las 16:33 solo tenemos el script, y a las 16:34 (como habíamos programado en Crontab) ya tenemos el backup del día 16.

```
pi@raspberrypi:~/backups $ date
Sun 16 May 16:33:40 BST 2021
pi@raspberrypi:~/backups $ ls -l
total 4
-rwx----- 1 pi pi 168 May 16 12:45 copia_seg_web.sh
pi@raspberrypi:~/backups $ date
Sun 16 May 16:34:05 BST 2021
pi@raspberrypi:~/backups $ ls -l
total 23140
-rw-r--r-- 1 pi www-data 23691264 May 16 16:34 backup_16-05-21.tgz
-rwx----- 1 pi pi 168 May 16 12:45 copia_seg_web.sh
pi@raspberrypi:~/backups $
```

También se ha realizado un trigger enfocado a realizar una copia de seguridad de la tabla reservas de la base de datos.

Su función es que cada vez que se inserte un registro en la tabla reservas (cuando se realice una reserva, por ejemplo), realiza el mismo Insert, en una tabla que hemos creado previamente (reservas.copia)

```
-- TRIGGER COPIA DE SEGURIDAD TABLA RESERVAS --  
DELIMITER $$  
• CREATE  
  TRIGGER insert_into_reservas AFTER INSERT ON reservas  
  FOR EACH ROW BEGIN  
    INSERT INTO reservas.copia (ID, usuario, nombre, apellidos, telefono, email, calle, calle_numero, ciudad, pais, fecha_entrada,  
    fecha_salida, personas, habitaciones, servicios) VALUES (new.ID, new.usuario, new.nombre, new.apellidos, new.telefono, new.email,  
    new.calle, new.calle_numero, new.ciudad, new.pais, new.fecha_entrada, new.fecha_salida, new.personas, new.habitaciones,  
    new.servicios);  
  END$$  
DELIMITER ;
```

Realizamos una reserva de prueba para comprobar que funciona:

Usuario: **pablo193**

REALIZA TU RESERVA

Nombre  
Probando

Apellidos  
COPIA SEGURIDAD

Teléfono  
690645

Email

Mostrando filas 0 - 0 (total de 1, La consulta tardó 0.0006 segundos.)

SELECT \* FROM 'reservas'

☐ Perfilando [Editar en línea] [Editar] [Explorar]

☐ Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla

+ Opciones

	ID	usuario	nombre	apellidos	telefono	email	calle	calle_numero	ciudad	codigo_postal	pais	fecha_entrada	fecha_salida	personas	habitaciones	servicios
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	22	pablo193	Probando	COPIA SEGURIDAD	690645	pablo193@gmail.com	AVDA AMERICA	24	León	24402	España	2021-05-19				

☐ Seleccionar todo | Para los elementos que están marcados: ☐ Editar ☐ Copiar ☐ Borrar ☐ Exportar

☐ Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla

Mostrando filas 0 - 0 (total de 1, La consulta tardó 0,0008 segundos.)

SELECT \* FROM 'reservas.copia'

Perfilando [Editar en línea] [Editar] [Explorar]

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla

+ Opciones

	ID	usuario	nombre	apellidos	telefono	email	calle	calle_numero	ciudad	codigo_postal	pais	fecha_entrada
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	22	pablobl93	Probando	COPIA SEGURIDAD	690645	pablobl93@gmail.com	AVDA AMERICA	24	León	0	España	2021-05-19

Seleccionar todo | Para los elementos que están marcados: ☐ Editar ☐ Copiar ☐ Borrar ☐ Exportar

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla

## 8. RECURSOS WEB

### 8.1 YOUTUBE

Hemos subido algunos videos a la plataforma YouTube para mostrar el funcionamiento de la web:

- [Registro y acceso de usuarios a Web SierraDeGata](#)
- [Registro y acceso de empleados a Web SierraDeGata](#)
- [Resto de páginas del sitio web](#)
- [SLIDER WEB](#)
- [Efecto FadeIn\(\) jQuery en Web](#)



Lista de reproducción con todos los vídeos: <https://bit.ly/3uhjYbE>

### 8.2 GITHUB











**GitHub** es un portal creado para alojar cualquier tipo de código de aplicaciones.

Se ha usado este portal para crear un repositorio, que actúa también, de copia de seguridad, donde se subirán los archivos para crear la página web, así como, un SQL con la creación de la base de datos y todas sus tablas y relaciones.












 [bl1993/HotelSierraDeGata-Backup](#)

 bbdd-sql
 web

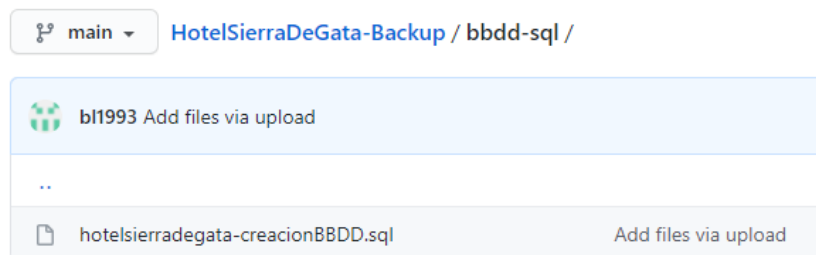
## Los archivos principales /web/

 main	HotelSierraDeGata-Backup / web /
 b11993 Add files via upload	
..	
 login	Add files via upload
 contacto.php	Add files via upload
 entorno.php	Add files via upload
 galeria.php	Add files via upload
 galeria2.php	Add files via upload
 habitaciones.php	Add files via upload
 historia.php	Add files via upload
 index.php	Add files via upload

## Carpeta /web/login/

 main	HotelSierraDeGata-Backup / web / login /
 b11993 Add files via upload	
..	
 acceso_empleados	Add files via upload
 acceso_usuarios	Add files via upload
 js	Add files via upload
 php	Add files via upload
 acceso.php	Add files via upload
 acceso_empleados.php	Add files via upload
 registro.php	Add files via upload
 scripts.php	Add files via upload
 styles.css	Add files via upload

**Carpeta /bbdd-sql/:** con archivo SQL de creación de base de datos y tablas.



## 9. CONCLUSIONES

Desde el punto de vista personal, me ha gustado realizar este proyecto porque me ha permitido poner en práctica los conocimientos teóricos y prácticos aprendidos durante estos dos cursos del ciclo formativo.

Además, he tenido que investigar acerca del hardware utilizado, ya que no tenía experiencia alguna en el manejo de Raspberry. También he realizado una búsqueda de información sobre varios otros temas que desconocía, como de las librerías o framework de PHP y JavaScript o DNS dinámico con No-IP.

Me hubiera gustado aplicar ciertas mejoras en el proyecto que tal vez por falta de tiempo no he podido realizar, como pueden ser:

### MEJORAR LA SEGURIDAD TANTO EN RASPBERRY COMO EN WEB

Por ejemplo, es un inconveniente que la web no esté accesible las 24 horas del día, por motivos de seguridad, preferimos no tener los puertos del router abiertos continuamente sin tener un sistema de seguridad fuerte que impida un posible ataque.

### FORMULARIO DE CONTACTO CON RESPUESTA

Es algo que me hubiera gustado realizar, el poder almacenar esos mensajes y poder leerlos y contestarlos, para ello, quizás deberíamos implementar un servidor de correo.

### INVESTIGAR SUBIDA DESDE TERMINAL A GITHUB

GitHub es una plataforma que se usa mucho en el entorno laboral, es por esto, que he querido iniciarme en la creación de repositorios, como hicimos con la creación de nuestro sitio en esta plataforma como copia de seguridad.

Investigando sobre Git, también vi la posibilidad de, usando comandos en la CLI de un sistema Linux, poder subir directorios o archivos a un repositorio que tengamos abierto en este portal como por ejemplo, *git add git commit git diff git stash .gitignore, etc.*

## 10. BIBLIOGRAFIA

Principales páginas consultadas:

- [StackOverflow](#): Para cualquier información o problema ocurrido en el día a día de la creación del proyecto.
- <https://pedropablomoral.com/> :Información sobre Raspberry y redes.
- [www.raulprietofernandez.net](http://www.raulprietofernandez.net): Información sobre redes.
- <https://www.ionos.es/digitalguide/servidores/know-how/que-es-un-dyndns-dns-dinamico/> : DNS DINÁMICO
- <https://www.xataka.com/ordenadores/raspberry-pi-3-model-b-analisis-mas-potencia-y-mejor-wifi-para-un-minipc-que-sigue-asombrando> Raspberry
- <https://manuales.guebs.com/mysql-5.0/triggers.html> Disparadores Triggers. -