

Memoria Sección 2_3

COW

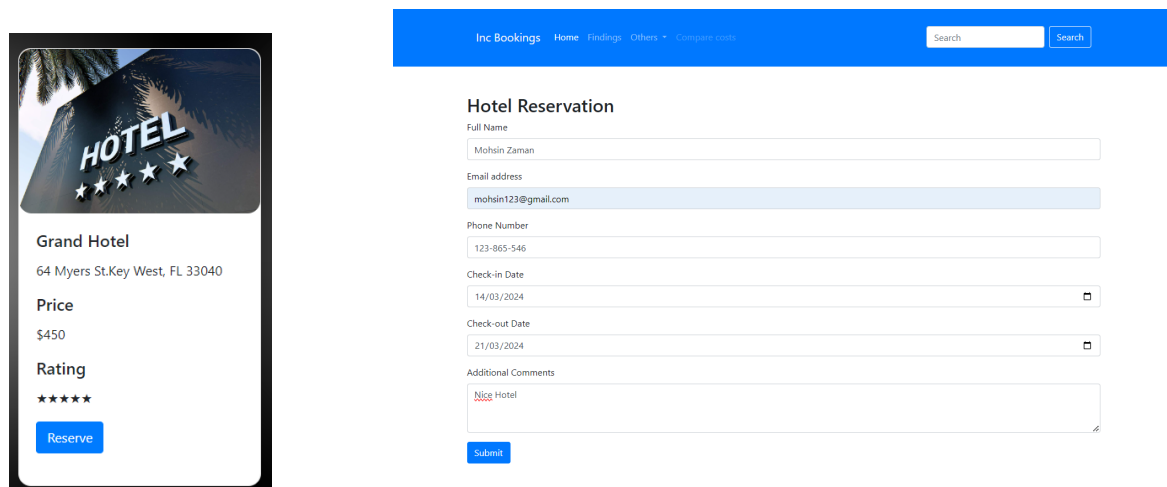
Nombre: Muhammad Mohsin Zaman Shaheen
Fecha: 17/3/2024

Apartado 1 Sección 2

En esta sección se pedía implementar dos páginas web PHP encargadas de reservar un hotel. Concretamente, se ha creado una página llamada *cliente.php* encargada de mostrar el formulario para reservar el hotel introduciendo los datos correspondientes. A fin de efectuar la reserva, se ha implementado el lado servidor en el fichero *server.php* para realizar las comprobaciones de los datos introducidos en el formulario y efectuar la reserva.

Así pues, el flujo para realizar la reserva es de la siguiente manera:

Desde la página principal, al clicar en el botón de reservar, aparece el formulario para introducir los datos. Para asegurar que los datos y sus formatos sean correctos, desde el lado cliente se han implementado expresiones regulares (*regexp*) para todos los campos de datos.



Control de datos lado cliente

Para el cambio de nombre, este no puede estar vacío lo cual se ha especificado por el atributo *required*:

```
<input type="text" class="form-control" id="name" name="name" required="required" placeholder="Enter your full name">
```

Para el e-mail, este debe seguir estrictamente el formato especificado en el campo pattern:

```
<input type="email" class="form-control" id="email" name="email" placeholder="Enter email" pattern="/^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/"/>
```

Para el campo de teléfono también he especificado un formato específico:

```
<input type="tel" class="form-control" id="phone" name="phone" placeholder="Enter phone number" pattern="[0-9]{3}-[0-9]{3}-[0-9]{3}"/>
```

Para el campo de comentario, siguiendo la referencia de las transparencias de teoría, he limitado a un máximo de 150 caracteres:

```
<textarea class="form-control" id="comments" name="comments" rows="3" placeholder="Enter any additional comments" maxlength="150"></textarea>
```

Pruebas

Para comprobar que todo funcione como esperaba, he realizado varias pruebas, una de las cuales es la siguiente donde el teléfono se ha introducido con un formato incorrecto:

Hotel Reservation

Full Name

Mohsin Zaman

Email address

mohsin123@gmail.com

Phone Number

1566-556-782

Check-in Date

12/03/2024

Check-out Date

22/03/2024

Additional Comments

My comment

Submit

Utiliza un formato que coincida con el solicitado
Phone number must be in the format: xxx-xxx-xxxx

Una vez que todos los datos cumplan el formato establecido, al clicar al botón de *submit* se muestra la página de confirmación mostrando todos los datos de la reserva. Para enviar los datos desde el formulario al servidor, utilizo el método POST inspirandome de las transparencias de teoría.

```
<form action="server.php" method="post">
```

Esto hace que al clicar en *submit* se pasen los datos al `server.php` que se encarga de leer los datos y otra vez realizar comprobaciones de datos desde lado servidor para mayor seguridad ya que desde el lado cliente no es 100% fiable. El servidor lee los datos utilizando **peticiones POST** y aparte de comprobar nombre, email y número de teléfono, también comprueba que en la fecha de reserva la fecha de salida no sea inferior a la de entrada. En caso que todo esté correcto redirige a la *página de confirmación* y en caso contrario, redirige a la *página de error*.

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {  
    $name = $_POST['name'];  
    $email = $_POST['email'];  
    $phone = $_POST['phone'];  
    $checkinDate = $_POST['checkinDate'];  
    $checkoutDate = $_POST['checkoutDate'];  
    $comments = $_POST['comments'];  
    $hotelName = $_POST['hotel'];  
    $price = $_POST['price'];  
}
```

Comprobación de los datos usando *regex* y métodos como *preg_match()* y *strlen()* inspirando de las transparencias de teoría:

```
// Validation of dates  
$checkin_timestamp = strtotime($checkinDate);  
$checkout_timestamp = strtotime($checkoutDate);  
  
// Check if the checkout date is before the check-in date  
if ($checkout_timestamp < $checkin_timestamp) {  
    $errors[] = "Checkout date cannot be before the check-in date.";  
} else {  
    echo "Checkout date is valid.";  
}
```

```
//Validation via REGEXP  
$email_regex = '/^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/';  
$phone_regex = '/^\d{3}-\d{3}-\d{3}$/';  
  
// Validate input  
$errors = [];  
if (strlen($name) < 1) {  
    $errors[] = "Name cannot be empty";  
}  
if (!preg_match($email_regex, $email)) {  
    $errors[] = "Invalid email format";  
}  
if (!preg_match($phone_regex, $phone)) {  
    $errors[] = "Invalid phone number format";  
}
```

Redireccionamiento a otros fichero:

Método: **header()**

```
// If there are no errors, proceed to display confirmation
if (empty($errors)) {
    // Display confirmation page
    header("Location: confirmation.php?hotel=$hotelName&price=$price&name=$name&email=$email&phone=$phone&checkin=$checkinDate&checkout=$checkoutDate");
    exit();
} else {
    // Display errors page
    header("Location: errors.php?hotel=$hotelName&price=$price&errors=" . urlencode(implode(", ", $errors)));
    exit();
}
```

En caso de reserva confirmada, se muestra la siguiente página. Para mostrar los datos en esta nueva página, utilizó **peticiones GET** para obtener los datos que ha pasado el servidor desde la URL como se muestra a continuación.

```
Thank you for your reservation, <?php echo $_GET['name']; ?>!<br>
Your reservation details:<br>
Hotel: <?php echo $_GET['hotel']; ?><br>
Paid Amount: <?php echo $_GET['price']; ?><br>
Email: <?php echo $_GET['email']; ?><br>
Phone: <?php echo $_GET['phone']; ?><br>
Check-in Date: <?php echo $_GET['checkin']; ?><br>
Check-out Date: <?php echo $_GET['checkout']; ?><br>
Comments: <?php echo $_GET['comments']; ?>
```

Confirmation

Thank you for your reservation, Mohsin Zaman!
Your reservation details:
Hotel: Grand Hotel
Paid Amount: \$450
Email: mohsin123@gmail.com
Phone: 156-556-782
Check-in Date: 2024-03-12
Check-out Date: 2024-03-22
Comments: My comment.

[Back to Main Page](#)

Finalmente, en caso que algún campo no cumpla con el formato o dato correcto, el servidor hace las comprobaciones y redirige a la página de error indicando cuál es el campo incorrecto

Hotel Reservation

Full Name

Email address

Phone Number

Check-in Date

Check-out Date

Additional Comments

Errors

Checkout date cannot be before the check-in date.
Please go back and correct the errors.

Apartado 2 Sección 3

En este apartado, el objetivo principal era interactuar con las bases de datos en phpmyadmin realizando operaciones de inserción/lectura desde el código php. Además, también he realizado operación de eliminación para extender más. Así pues, este objetivo se ha cumplido de la siguiente forma siguiendo todos los ejemplos proporcionados en las transparencias de teoría.

En esta sección la forma de reservar es la misma que en el apartado anterior pero la diferencia es que al clicar en el botón *submit*, aparte de mostrar la página de confirmación, se guardan los datos del formulario en la base de datos. Cabe destacar que estoy usando la base de datos “world” pero se puede usar otra siempre y cuando se especifique las credenciales en el fichero *db_connection.php*.

```
<?php
// Database credentials
$servername = "localhost";
$username = "root";
$password = "";
$dbase = "world";

try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbase", $username, $password);
    // Set PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e) {
    echo "Connection failed: " . $e->getMessage();
}
?>
```

require() y requiere_once

He aprovechado el fichero *db_connection.php* en diferentes ficheros usando métodos como *require()* en *server.php* y *requiere_once* en *paginaBootstrap.php*.

```
require('../db_management/db_connection.php');

if ($_SERVER["REQUEST_METHOD"] == "POST") {
```

```
<div class="row">
    <?php
    require_once '../php/db_management/db_connection.php';
```

Para guardar los datos en la base de datos, se crea la tabla “reservations” en caso que no exista previamente. Para comprobar si la tabla existe, utilizo métodos como *prepare()* y *fetchColumn()* como se ha visto en teoría.

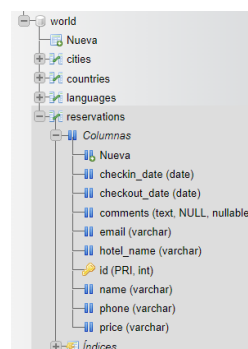
```
// Check if the reservations table exists

$stmt = $conn->prepare("SELECT 1 FROM information_schema.tables WHERE table_schema = ? AND table_name = ?");
$stmt->execute([$dbase, 'reservations']);
$tableExists = ($stmt->fetchColumn() !== false);
```

CREATE TABLE / INSERT TO DATABASE

Si no existe la tabla, la creamos desde el código:

```
if (!$tableExists) {
    // Create the table
    $reser_table = "CREATE TABLE reservations (
        id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
        name VARCHAR(30) NOT NULL,
        email VARCHAR(50) NOT NULL,
        phone VARCHAR(15) NOT NULL,
        checkin_date DATE NOT NULL,
        checkout_date DATE NOT NULL,
        comments TEXT,
        hotel_name VARCHAR(50) NOT NULL,
        price VARCHAR(6) NOT NULL
    )";
    $conn->exec($reser_table);
    echo "Table MyGuests created successfully";
}
```



Y insertamos datos en la tabla y finalmente cerramos la conexión poniendo “*\$conn = null*” inspirando de los ejemplos de teoría.


```
// Just insert data into the table
$insert = "INSERT INTO reservations (name, email, phone, checkin_date, checkout_date, comments, hotel_name, price) VALUES ('$name', '$email', '$phone', '$checki
$conn->exec($insert);
echo "New record created successfully";
$conn = null;
```

	id	name	email	phone	checkin_date	checkout_date	comments	hotel_name	price
<input type="checkbox"/>	5	Mohsin Zaman	mohsin123@gmail.com	325-886-471	2024-03-13	2024-03-22	Reserving my hotel!	PaletteParadise Stays	\$215


READING FROM DATABASE

Por tal de realizar la lectura de los datos desde la base de datos, en vez de leer otros datos randoms de las otras bases de datos, he preferido añadir una funcionalidad nueva a mi pagina web que trata de mostrar los hoteles como “reservado” y con opción a cancelar reserva y de esta forma, realizo operaciones de lectura y eliminación en la base de datos.


Un hotel reservado se muestra tal que:



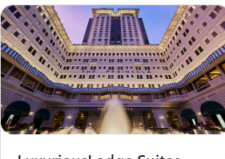
Hotel Bellevue
456 Park Avenue, Los Angeles, USA
Price
\$345
Rating
★★★★☆
[Reserve](#)



Grand Hotel
64 Myers St.Key West, FL 33040
Price
\$450
Rating
★★★★★
[Reserve](#)



PaletteParadise Stays
456 Park Avenue, Los Angeles, USA
Price
\$215
Rating
★★★★☆
[Reserved](#) [Cancel Reservation](#)



LuxuriousLodge Suites
64 Cobblestone Drive Bethlehem, PA 18015 USA
Price
\$749
Rating
★★★★☆
[Reserve](#)

Para hacer esto, he realizado lectura en la tabla “reservations” de la siguiente manera en el fichero paginaBootstrap.php y si el hotel esta en la base de datos, se marca como reservado.

```
function isHotelReserved($hotelName,$conn) {
    $result = $conn->query("SELECT * FROM reservations WHERE hotel_name = '$hotelName' ");
    return $result->rowCount() > 0;
}
```

DELETE FROM DATABASE

Esta operación se realiza si se quiere cancelar una reserva clicando al botón rojo de “Cancel Reservation”. Al clicar, se ejecuta el *cancel_reservation.php* que elimina la reserva de la base de datos y el hotel vuelve a estar disponible para reservar en la página principal.

```
<form action="./php/reservation_handling/cancel_reservation.php" method="post">
    <input type="hidden" name="hotelName" value="<?= $hotel['name'] ?>">
    <button type="submit" class="btn btn-danger">Cancel Reservation</button>
</form>
```

```
try {
    // Escape and quote the hotel name
    $quotedHotelName = $conn->quote($hotelName);
    $sql = "DELETE FROM reservations WHERE hotel_name = $quotedHotelName";
    $affectedRows = $conn->exec($sql);
    if ($affectedRows > 0) {
        header("Location: reservation_canceled.php?hotel=$hotelName");
        exit();
    } else {
        echo "No hotel found with the provided name.";
    }
} catch (PDOException $e) {
    echo "Error: " . $e->getMessage();
}
```

Confirmation

Your reservation for the following hotel has ben canceled sucefully:
Hotel: PaletteParadise Stays

[Back to Main Page](#)