### \*\*Actualización de la Documentación de la API\*\*

Con la información proporcionada sobre `config/database.php` y `helpers/utils.php`, voy a actualizar la documentación para incluir detalles sobre la configuración de la base de datos, las funciones auxiliares y su integración con los endpoints. Esto permitirá una comprensión más completa de cómo está estructurada la API y cómo interactúan sus componentes.

---

## \*\*Documentación de la API\*\*

#### \*\*Introducción\*\*

La siguiente documentación describe los endpoints de una API RESTful desarrollada en PHP puro, conectada a un frontend en React. La API gestiona funcionalidades relacionadas con reservas de salones, testimonios y detalles del salón. Además, se incluyen detalles sobre la configuración de la base de datos y las funciones auxiliares utilizadas.

---

### \*\*Estructura General\*\*

#### \*\*Componentes principales\*\*

1. \*\*Base de Datos\*\*:

- Configurada en `config/database.php` utilizando PDO para conexiones seguras.

2. \*\*Funciones Auxiliares\*\*:

- Implementadas en `helpers/utils.php` para manejar CORS, sanitización de entradas y respuestas JSON.

3. \*\*Endpoints\*\*:

- `/api/v1/nuevotestimonio.php`: Crear nuevos testimonios.

- `/api/v1/reservas.php`: Gestionar reservas (crear, verificar disponibilidad, obtener fechas ocupadas).

- `/api/v1/salon.php`: Obtener detalles del salón y sus imágenes de galería.

- `/api/v1/testimonios.php`: Manejar testimonios (obtener aprobados y enviar nuevos).

---

### \*\*Configuración de la Base de Datos\*\*

#### \*\*Archivo: `config/database.php`\*\*

Este archivo contiene la clase `Database` que maneja la conexión a la base de datos utilizando PDO. Proporciona métodos para conectar de forma segura y manejar errores.

##### \*\*Propiedades de la clase `Database`:\*\*

| Propiedad | Descripción |

|--------------|-----------------------------------------------------------------------------|

| `$host` | Host de la base de datos (generalmente `localhost` para entornos locales). |

| `$db\_name` | Nombre de la base de datos (`salon\_eventos`). |

| `$username` | Usuario de la base de datos (por defecto `root` en XAMPP). |

| `$password` | Contraseña de la base de datos (por defecto vacía en XAMPP). |

##### \*\*Método `getConnection()`:\*\*

- \*\*Función\*\*: Establece una conexión a la base de datos utilizando PDO.

- \*\*Configuración de DSN\*\*:

```php

$dsn = "mysql:host=" . $this->host . ";dbname=" . $this->db\_name . ";charset=utf8mb4";

```

- \*\*Opciones de PDO\*\*:

```php

$options = [

PDO::ATTR\_ERRMODE => PDO::ERRMODE\_EXCEPTION, // Lanza excepciones en caso de error

PDO::ATTR\_DEFAULT\_FETCH\_MODE => PDO::FETCH\_ASSOC, // Resultados como arrays asociativos

PDO::ATTR\_EMULATE\_PREPARES => false, // Desactiva la emulación de preparaciones para mayor seguridad

];

```

- \*\*Manejo de errores\*\*:

- Si ocurre un error al conectar, se registra el mensaje en el log y se envía una respuesta HTTP 500 con un mensaje genérico.

##### \*\*Uso en los endpoints\*\*:

Cada endpoint utiliza la clase `Database` para obtener una conexión a la base de datos:

```php

require\_once '../../config/database.php';

$db = (new Database())->getConnection();

```

---

### \*\*Funciones Auxiliares\*\*

#### \*\*Archivo: `helpers/utils.php`\*\*

Este archivo contiene funciones reutilizables para manejar CORS, sanitizar entradas y enviar respuestas JSON.

##### \*\*Función `setCorsHeaders()`\*\*:

- \*\*Función\*\*: Configura los encabezados CORS para permitir solicitudes desde el frontend.

- \*\*Encabezados configurados\*\*:

```php

header("Access-Control-Allow-Origin: http://localhost:3000");

header("Content-Type: application/json; charset=UTF-8");

header("Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS");

header("Access-Control-Max-Age: 3600");

header("Access-Control-Allow-Headers: Content-Type, Access-Control-Allow-Headers, Authorization, X-Requested-With");

```

- \*\*Manejo de solicitudes OPTIONS\*\*:

- Si el método de solicitud es `OPTIONS`, se responde con un código 200 y se termina la ejecución.

##### \*\*Función `sanitizeInput($data)`\*\*:

- \*\*Función\*\*: Sanitiza una cadena de texto para prevenir inyecciones SQL y ataques XSS.

- \*\*Pasos\*\*:

1. Elimina espacios en blanco (`trim`).

2. Elimina barras invertidas (`stripslashes`).

3. Convierte caracteres especiales a entidades HTML (`htmlspecialchars`).

##### \*\*Función `sendJsonResponse($data, $statusCode)`\*\*:

- \*\*Función\*\*: Envía una respuesta JSON al cliente y termina la ejecución.

- \*\*Parámetros\*\*:

- `$data`: Los datos a enviar en formato JSON.

- `$statusCode`: El código de estado HTTP (por defecto 200).

- \*\*Ejemplo de uso\*\*:

```php

sendJsonResponse(["message" => "Reserva creada exitosamente."], 201);

```

---

### \*\*Endpoints Detallados\*\*

#### \*\*1. `GET /api/v1/testimonios.php`\*\*

##### \*\*Función:\*\*

Obtiene todos los testimonios aprobados ordenados por fecha de creación descendente.

##### \*\*Método HTTP:\*\*

`GET`

##### \*\*URL:\*\*

`/api/v1/testimonios.php`

##### \*\*Parámetros de consulta:\*\*

Ninguno.

##### \*\*Códigos de respuesta:\*\*

- \*\*200 OK\*\*: Se devuelven los testimonios aprobados.

```json

[

{

"name": "Juan Pérez",

"comment": "Excelente servicio!",

"rating": 5

},

{

"name": "María López",

"comment": "Muy buen ambiente.",

"rating": 4

}

]

```

---

#### \*\*2. `POST /api/v1/testimonios.php`\*\*

##### \*\*Función:\*\*

Permite crear un nuevo testimonio en la base de datos. Los testimonios requieren moderación antes de ser visibles públicamente.

##### \*\*Método HTTP:\*\*

`POST`

##### \*\*URL:\*\*

`/api/v1/testimonios.php`

##### \*\*Parámetros de solicitud:\*\*

Los datos se envían en el cuerpo de la solicitud (`Content-Type: application/json`).

| Nombre del parámetro | Tipo | Descripción | Requerido |

|----------------------|--------|--------------------------------------|-----------|

| `nombre\_cliente` | string | Nombre del cliente que deja el testimonio | Sí |

| `comentario` | string | Comentario del cliente | Sí |

| `calificacion` | int | Calificación (entre 1 y 5) | Sí |

##### \*\*Códigos de respuesta:\*\*

- \*\*201 Created\*\*: El testimonio se ha enviado correctamente para revisión.

```json

{

"status": "success",

"message": "Testimonio enviado para revisión. ¡Gracias!"

}

```

---

#### \*\*3. `POST /api/v1/reservas.php?action=checkAvailability`\*\*

##### \*\*Función:\*\*

Verifica la disponibilidad de un rango de fechas para reservar el salón.

##### \*\*Método HTTP:\*\*

`GET`

##### \*\*URL:\*\*

`/api/v1/reservas.php?action=checkAvailability`

##### \*\*Parámetros de consulta:\*\*

| Nombre del parámetro | Tipo | Descripción | Requerido |

|----------------------|--------|--------------------------------------|-----------|

| `fecha\_inicio` | string | Fecha de inicio del rango (formato YYYY-MM-DD) | Sí |

| `fecha\_fin` | string | Fecha de fin del rango (formato YYYY-MM-DD) | Sí |

##### \*\*Códigos de respuesta:\*\*

- \*\*200 OK\*\*: Las fechas están disponibles.

```json

{

"disponible": true,

"message": "Fechas disponibles."

}

```

---

#### \*\*4. `GET /api/v1/salon.php`\*\*

##### \*\*Función:\*\*

Obtiene los detalles del salón y sus imágenes de galería.

##### \*\*Método HTTP:\*\*

`GET`

##### \*\*URL:\*\*

`/api/v1/salon.php`

##### \*\*Parámetros de consulta:\*\*

| Nombre del parámetro | Tipo | Descripción | Requerido |

|----------------------|--------|--------------------------------------|-----------|

| `action` | string | Acción a realizar (`getGalleryImages` o vacío) | No |

##### \*\*Códigos de respuesta:\*\*

- \*\*200 OK\*\*: Se devuelven los detalles del salón o las imágenes de la galería.

- Para `action=getGalleryImages`:

```json

[

{

"url\_imagen": "https://example.com/image1.jpg",

"alt\_text": "Imagen 1"

},

...

]

```

---

### \*\*Conclusión\*\*

Esta documentación ahora incluye detalles completos sobre la configuración de la base de datos (`config/database.php`) y las funciones auxiliares (`helpers/utils.php`). También se describen todos los endpoints disponibles en la API, sus funciones, parámetros y respuestas esperadas.