PREGUNTAS TEÓRICAS TIPO EXAMEN - 1 HORA

Basadas en los ejercicios del curso

BLOQUE 1: ARQUITECTURAS WEB (del ejercicio UD1)

Pregunta 1

¿Qué tipo de páginas (estáticas o dinámicas) utilizarías para cada una de estas páginas de una aplicación de gestión de correos?

- **Página de presentación:** ESTÁTICA (o dinámica simple) - el contenido no cambia, solo muestra información fija.
- **Página de introducción de datos (formulario):** DINÁMICA - debe procesar el formulario y guardar datos.
- **Página de visualización:** DINÁMICA - lee datos de la BBDD y los muestra.

Pregunta 2

Si quieres validar que un email tiene @ ANTES de enviar el formulario, ¿qué tecnología usas?

JavaScript (lado cliente). Se ejecuta en el navegador antes de enviar los datos al servidor.

Pregunta 3

Si quieres comprobar que un email NO existe ya en la base de datos, ¿qué tecnología usas?

PHP (lado servidor). Necesitas acceder a la BBDD, que solo está disponible desde el servidor.

Pregunta 4

¿Qué arquitecturas puedes usar en el servidor? ¿Qué tipo de lenguaje usa cada una?

Arquitectura	Lenguaje	Tipo
LAMP/WAMP	PHP	Lenguaje de guiones (interpretado)
Java EE	Java	Compilado a código intermedio (bytecode)
.NET	C#	Compilado a código intermedio (MSIL)
Node.js	JavaScript	Lenguaje de guiones (interpretado)

Pregunta 5

¿Qué parámetros debes considerar para elegir una arquitectura?

- Rendimiento necesario
- Coste de licencias
- Conocimientos del equipo

- Escalabilidad requerida
- Compatibilidad con sistemas existentes
- Comunidad y soporte disponible

Pregunta 6

¿Qué componentes necesita una arquitectura LAMP?

Componente	Función	Ejemplo concreto
L - Linux	Sistema operativo	Ubuntu, Debian
A - Apache	Servidor web	Apache HTTP Server
M - MySQL	Base de datos	MySQL, MariaDB
P - PHP	Lenguaje servidor	PHP 8.x

Pregunta 7

¿Qué necesitas instalar para desarrollar una aplicación PHP?

- Editor de código (VS Code, PHPStorm)
- Servidor web (Apache o Nginx)
- Intérprete PHP
- Base de datos (MySQL/MariaDB)
- Gestor de dependencias (Composer)
- Navegador para pruebas

Pregunta 8

Si usas PHP, ¿qué tipo de dato se usa para una dirección de correo?

string - Las direcciones de correo se almacenan como cadenas de texto.

BLOQUE 2: FORMULARIOS Y SUPERGLOBALES

Pregunta 9

¿Cuál es la diferencia entre $_GET y $_POST?

$_GET	$_POST
Datos visibles en URL	Datos ocultos en cuerpo de petición
Límite de longitud (~2000 chars)	Sin límite práctico
Se puede marcar como favorito	No se puede marcar
Para consultas, búsquedas	Para envío de datos sensibles

Pregunta 10

¿Qué contiene $_SERVER['REQUEST_METHOD']?

Contiene el método HTTP de la petición actual: `"GET"` o `"POST"` (u otros como PUT, DELETE).

Uso típico:

```
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Procesar formulario enviado por POST
}
```

Pregunta 11

¿Qué atributo debe tener un formulario para subir archivos?

```
<form method="POST" enctype="multipart/form-data">
```

El atributo `enctype="multipart/form-data"` es **obligatorio** para subir archivos.

Pregunta 12

¿Qué información contiene $_FILES['archivo']?

```
$_FILES['archivo']['name']     // Nombre original del archivo
$_FILES['archivo']['type']     // Tipo MIME (ej: application/pdf)
$_FILES['archivo']['tmp_name'] // Ruta temporal en servidor
$_FILES['archivo']['size']     // Tamaño en bytes
$_FILES['archivo']['error']    // Código de error (0 = OK)
```

Pregunta 13

¿Cómo validas que un archivo sea PDF?

Debes validar **DOS cosas**:

```
// 1. Extensión del archivo
$extension = pathinfo($_FILES['archivo']['name'], PATHINFO_EXTENSION);
$extensionOk = strtolower($extension) === 'pdf';

// 2. Tipo MIME
$tipoOk = $_FILES['archivo']['type'] === 'application/pdf';

// Ambos deben cumplirse
if ($extensionOk && $tipoOk) {
    // Es un PDF válido
}
```

Pregunta 14

¿Para qué sirve `file_exists()` y `move_uploaded_file()`?

- **file_exists($path)**: Comprueba si un archivo ya existe en esa ruta
- **move_uploaded_file($tmp, $destino)**: Mueve el archivo de la carpeta temporal a su destino final

---

BLOQUE 3: COOKIES

Pregunta 15

¿Qué es una cookie y dónde se almacena?

Una cookie es un pequeño archivo de texto que se almacena en el **navegador del cliente**. Permite guardar información entre peticiones HTTP.

Pregunta 16

¿Cuál es la sintaxis para crear una cookie?

```
setcookie("nombre", "valor", time() + segundos);
```

Ejemplos de duración:

- 1 hora: `time() + 3600`
- 1 día: `time() + 86400`
- 7 días: `time() + 604800`

Pregunta 17

¿Por qué setcookie() debe ir ANTES de cualquier echo o HTML?

Porque las cookies se envían en las **cabeceras HTTP**. Una vez que se ha enviado contenido (HTML, echo, espacios en blanco), las cabeceras ya se han enviado y no se pueden modificar.

Pregunta 18

¿Cómo se elimina una cookie?

```
setcookie("nombre", "", time() - 3600);
```

Se establece con un tiempo de expiración **en el pasado**.

Pregunta 19

¿Cómo lees una cookie?

```php
if (isset($_COOKIE['nombre'])) {
    $valor = $_COOKIE['nombre'];
}
```

Importante: Las cookies NO están disponibles en la misma petición en que se crean. Estarán disponibles en la **siguiente petición**.

BLOQUE 4: SESIONES

Pregunta 20

¿Cuál es la diferencia entre cookies y sesiones?

Cookies	Sesiones
Se guardan en el cliente	Se guardan en el servidor
Tamaño limitado (~4KB)	Sin límite práctico
Visible/editable por usuario	Seguras, no accesibles
Persisten tras cerrar navegador	Se pierden al cerrar (por defecto)

Pregunta 21

¿Qué hace session_start()?

- Inicia o reanuda una sesión existente
- Crea la variable superglobal $_SESSION
- Genera un ID de sesión único (PHPSESSID)
- **Debe llamarse al principio de cada script** que use sesiones

Pregunta 22

¿Cómo cierras una sesión completamente?

```php
session_start();          // Primero iniciarla
$_SESSION = [];           // O session_unset()
session_destroy();        // Destruir datos del servidor
setcookie(session_name(), '', time() - 3600);  // Eliminar cookie
```

Pregunta 23

¿Para qué sirve `header('Location: ...')`?

Para **redirigir** al navegador a otra página.

Importante: Siempre debe ir seguido de `exit;` para detener la ejecución del script actual.

```
header('Location: panel.php');
exit;
```

BLOQUE 5: BASE DE DATOS Y PDO

Pregunta 24

¿Qué es PDO y qué ventajas tiene?

PDO (PHP Data Objects) es una capa de abstracción para acceso a bases de datos.

Ventajas:

- Funciona con múltiples BBDD (MySQL, PostgreSQL, SQLite...)
- Sentencias preparadas (previene SQL Injection)
- Manejo de excepciones
- API orientada a objetos

Pregunta 25

¿Por qué se usan sentencias preparadas?

1. **Seguridad**: Previenen inyección SQL
2. **Rendimiento**: La consulta se compila una vez y se ejecuta múltiples veces
3. **Claridad**: Separan SQL de los datos

```
// MAL - Vulnerable a SQL Injection
$sql = "SELECT * FROM users WHERE id = " . $_GET['id'];

// BIEN - Seguro
$sql = "SELECT * FROM users WHERE id = :id";
$stmt = $pdo->prepare($sql);
$stmt->execute([':id' => $_GET['id']]);
```

Pregunta 26

¿Qué hace PDO::ERRMODE_EXCEPTION?

Configura PDO para que lance **excepciones** cuando ocurre un error SQL. Esto permite capturarlas con try-catch en lugar de tener que comprobar el resultado de cada operación.

Pregunta 27

¿Qué diferencia hay entre fetch() y fetchAll()?

- **fetch()**: Devuelve UNA fila (o false si no hay más)
- **fetchAll()**: Devuelve TODAS las filas en un array

```
$stmt->execute();
$todos = $stmt->fetchAll(PDO::FETCH_OBJ);  // Array de objetos
$uno = $stmt->fetch(PDO::FETCH_OBJ);       // Un objeto
```

BLOQUE 6: TRANSACCIONES

Pregunta 28

¿Qué es una transacción y para qué sirve?

Una transacción agrupa varias operaciones SQL en una **unidad atómica**: o se ejecutan TODAS correctamente, o no se ejecuta NINGUNA.

Propiedades ACID:

- **Atomicidad**: Todo o nada
- **Consistencia**: La BBDD queda en estado válido
- **Aislamiento**: Operaciones invisibles hasta commit
- **Durabilidad**: Cambios permanentes tras commit

Pregunta 29

¿Cuáles son los 3 métodos de PDO para transacciones?

```
$pdo->beginTransaction();  // Inicia la transacción
$pdo->commit();            // Confirma los cambios
$pdo->rollBack();          // Deshace los cambios
```

Pregunta 30

¿Por qué la tabla debe ser InnoDB?

Porque **MyISAM no soporta transacciones**. Solo el motor InnoDB permite usar beginTransaction, commit y rollBack.

Pregunta 31

Escribe el esqueleto de una transacción:

```
try {
    $pdo->beginTransaction();

    // Operaciones...
    $stmt = $pdo->prepare("INSERT...");
    $stmt->execute([...]);
```

```
    $pdo->commit();

} catch (PDOException $e) {
    $pdo->rollBack();
    echo "Error: " . $e->getMessage();
}
```

BLOQUE 7: PHPMAILER

Pregunta 32

¿Qué es Composer y para qué se usa?

Composer es el gestor de dependencias de PHP. Permite:

- Instalar librerías externas (como PHPMailer)
- Gestionar autoload de clases
- Definir namespaces con PSR-4

Pregunta 33

¿Cómo se instala PHPMailer?

```
composer require phpmailer/phpmailer
```

Pregunta 34

¿Qué configuración necesitas para Gmail?

```
$mail->Host = 'smtp.gmail.com';
$mail->Port = 587;
$mail->SMTPSecure = PHPMailer::ENCRYPTION_STARTTLS;
$mail->Username = 'tu@gmail.com';
$mail->Password = 'contraseña_de_aplicacion';  // No la normal
```

Nota: Necesitas una "contraseña de aplicación" de Google, no tu contraseña normal.

BLOQUE 8: POO EN PHP

Pregunta 35

¿Qué diferencia hay entre require, require_once, include e include_once?

Función	Si no existe el archivo	Si ya está incluido

Función	Si no existe el archivo	Si ya está incluido
require	Error FATAL (para script)	Lo incluye de nuevo
require_once	Error FATAL	NO lo incluye otra vez
include	Warning (continúa)	Lo incluye de nuevo
include_once	Warning	NO lo incluye otra vez

Pregunta 36

¿Qué es un namespace y para qué sirve?

Un **namespace** es una forma de organizar clases y evitar conflictos de nombres. Funciona como "carpetas" para las clases.

```php
namespace App\Clases;

class Usuario { }
```

```php
use App\Clases\Usuario;
$u = new Usuario();
```

Pregunta 37

¿Qué hace serialize() y unserialize()?

- **serialize($objeto)**: Convierte un objeto en una cadena de texto que se puede almacenar
- **unserialize($cadena)**: Reconstruye el objeto original desde la cadena

Uso: Guardar objetos en sesiones, cookies, archivos o BBDD.

RESPUESTAS RÁPIDAS

Pregunta	Respuesta
¿Cliente o servidor para validar formato?	Cliente (JavaScript)
¿Cliente o servidor para comprobar BBDD?	Servidor (PHP)
¿Dónde se guardan las cookies?	Navegador del cliente
¿Dónde se guardan las sesiones?	Servidor
¿Qué motor MySQL soporta transacciones?	InnoDB
¿Qué previenen las sentencias preparadas?	SQL Injection
¿Qué debe ir siempre después de header()?	exit;

Pregunta	Respuesta
¿Qué atributo para subir archivos?	enctype="multipart/form-data"
¿Puerto SMTP para Gmail?	587
¿Qué tipo PHP para un email?	string