ESTÁNDARES DE PROGRAMACIÓN

Estándares de programación de PHP

- Se debe implementar consultas preparadas para evitar las vulnerabilidades de inyección SQL, esto implica el uso de la clase MYSQL o PDO para preparar y ejecutar la consulta.
- Todos los datos recibidos desde el cliente, tales como `$_GET`, `$_POST` o `$_REQUEST`, son validados y sanitizados antes de ser utilizados en el código. Se emplea `filter_input()`.
- Para evitar vulnerabilidades XSS (Cross-Site Scripting), utiliza `htmlspecialchars()` al mostrar datos.
- Usar `isset()` para verificar la existencia de variables
- Organizar el código PHP de tal manera que la lógica de negocio se mantiene separada del código de representación HTML, se debe aplicar MVC Model-View-Controller.
- Definir variables que reflejen su propósito y evitar usar variables genéricas.

Estándares de programación de JavaScript

- Utilizar var para la declaración de variables en primera instancia.
- Implementar funciones flecas (=>) para reducir la verbosidad del código y mejorar la legibilidad, especialmente en los callbacks.
- Manejar eventos con jQuery, utilizando selectores específicos para evitar colisiones de eventos. Además, se asegura que cada evento tenga un namespace adecuado cuando sea necesario, evitando conflictos indeseados.

Estándares de programación en SQL

- Utiliza nombres descriptivos y sigue un estándar de nomenclatura consistente, como el uso de **snake_case** (product_name, category_id) o **camelCase** (productName, categoryId).
- Asegúrate de que los campos utilizados en los WHERE, ORDER BY y JOIN tengan índices para optimizar la consulta.
- Utilizar nombres descriptivos y claros, prefijos y sufijos consistentes.
- Usar claves primarias(id) en todas las tablas.
- Utilizar campos de timestamp para seguimiento (data_created, data_updated).

ESTÁNDARES DE PROGRAMACIÓN

`Estructura General de la creación de Tablas:`

```
CREATE TABLE nombre_tabla (
    id INT(30) NOT NULL AUTO_INCREMENT,
    -- Campos específicos de la entidad
    campo1 TIPO_DATO RESTRICCIONES,
    campo2 TIPO_DATO RESTRICCIONES,
    -- Campos de control y auditoría
    status TINYINT(1) NOT NULL DEFAULT 1,
    date_created DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    date_updated DATETIME DEFAULT NULL ON UPDATE
CURRENT_TIMESTAMP,
    -- Definición de clave primaria
    PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_general_ci;
```

Nomenclatura:

- Usar `snake_case` para nombres de tablas
- Nombres en inglés y descriptivos
- Usar plural para tablas que representan colecciones

Tipos de Datos:

- Usar tipos de datos precisos
- `INT(30)` para identificadores
- `VARCHAR` para cadenas con longitud limitada
- `TEXT` para descripciones largas
- `DOUBLE o FLOAT` para valores decimales
- `DATETIME` para fechas y timestamps

Campos Estándar:

- `id`: Identificador único
- `status`: Estado del registro (activo/inactivo)
- `date_created`: Fecha de creación
- `date_updated`: Fecha de última actualización

ESTÁNDARES DE PROGRAMACIÓN

Restricciones:

- `NOT NULL` para campos obligatorios
- `DEFAULT` para valores por defecto
- `AUTO_INCREMENT` para ids
- Definir relaciones con `FOREIGN KEY`