Resumen del Desarrollo - 13/04/2025

1. Guardar Estado de Ánimo (AJAX + PHP + MVC)

- Se captura el evento submit del formulario con JavaScript (emociones.js).
- Se envían los datos (estado de ánimo y texto) al backend mediante fetch POST (formato JSON).
- En moodController.php se reciben los datos y se llaman a las funciones del modelo.
- En el modelo EstadoAnimo.php, se ejecuta una consulta INSERT usando PDO para guardar los datos.
- Se responde con un mensaje JSON y se muestra un mensaje de confirmación al usuario.
- También se limpian los campos del formulario tras unos segundos.

2. Mostrar Historial de Registros

- Se creó un botón con id 'btn-ver-historial' que activa un fetch al backend.
- El archivo historial.js maneja ese evento y envía una acción 'ver_historial'.
- moodController.php evalúa la acción y llama al modelo para recuperar los registros del usuario.
- Se devuelve un array JSON con los datos.
- JavaScript crea dinámicamente elementos <li> con cada registro y los inyecta en el contenedor HTML.

3. Estructura del Proyecto

- Vistas: contiene los archivos PHP como dashboard.php y carpetas de CSS.
- assets/js: contiene los scripts JavaScript separados por funcionalidad (emociones.js, historial.js).
- controladores: donde se encuentra moodController.php.
- modelos: contiene EstadoAnimo.php con funciones guardar() e historial().

4. Buenas Prácticas Aplicadas

- Código JavaScript modularizado: separados por funcionalidades.
- Uso de MVC para estructurar backend.
- Validación de datos tanto en frontend como en backend.
- Respuestas en formato JSON para comunicación limpia.
- Limpieza de campos y feedback visual al usuario.

5. Detalles Técnicos Relevantes

Resumen del Desarrollo - 13/04/2025

- Se usó fetch con Content-Type 'application/json'.

- Se capturan errores con try/catch y bloques .catch() en JavaScript.

- Uso de PDO con prepared statements para seguridad en las consultas.

- Las respuestas del servidor se muestran al usuario mediante manipulación del DOM.