🔄 FLUJO ACTUAL DE LA APLICACIÓN DE EMOCIONES

📋 ÍNDICE

🎯 RESUMEN GENERAL

Propósito de la Aplicación

Sistema para registrar el estado emocional de empleados mediante un formulario interactivo, optimizado para uso en tablet en la entrada de una empresa.

Características Principales

- ☑ Formulario público sin autenticación
- ☑ Interfaz multilingüe (FR, EN, ES)
- ☑ Diseño responsive (móvil, tablet, desktop)
- ☑ Efectos visuales y animaciones
- ☑ Navegación por teclado
- ☑ Mensajes motivacionales dinámicos

🏗 ARQUITECTURA DEL SISTEMA

Tecnologías Utilizadas

- **Backend:** Laravel 10 (PHP)
- **Frontend:** Blade Templates + Tailwind CSS
- **JavaScript:** Vanilla JS + Chart.js (para futuros gráficos)
- **Base de Datos:** MySQL
- **Servidor:** XAMPP (Apache + MySQL)

Estructura de Archivos

```
gbnEmotions/
├── app/
```

```
│   ├── Http/Controllers/
│   │   └── MoodEmotionController.php
│   └── Models/
│       └── MoodEmotion.php
├── resources/
│   ├── views/
│   │   ├── moods/
│   │   │   └── create.blade.php
│   │   └── components/
│   │       └── language-selector.blade.php
│   └── css/
│       └── app.css
├── public/
│   └── js/
│       └── mood-form.js
├── lang/
│   ├── fr/moods.php
│   ├── en/moods.php
│   └── es/moods.php
├── routes/
│   └── web.php
└── database/
    └── migrations/
        └── mood_emotions_table.php
```

FLUJO DE USUARIO

Paso 1: Acceso al Formulario

```
1. Empleado accede a: /moods/{employee_id}/create
   Ejemplo: /moods/E001/create

2. Sistema muestra:
   - Título personalizado: "Registrar el ánimo del empleado E001"
   - Saludo según hora del día
   - Selector de idioma (FR/EN/ES)
   - 5 emojis de emociones
   - Indicador de progreso (Paso 1/2)
```

Paso 2: Selección de Emoción

```
1. Usuario hace clic en un emoji o navega con teclado
2. Sistema responde:
   - Efecto visual de pulso en emoji seleccionado
   - Cambio de color según emoción
   - Actualización del indicador de progreso (Paso 2/2)
   - Aparición de mensaje motivacional
   - Revelación de preguntas específicas
```

Paso 3: Respuesta a Preguntas

```
1. Sistema muestra 3 preguntas específicas según la emoción
2. Usuario selecciona Sí/No para cada pregunta
3. Preguntas aparecen con animación fade-in
4. Cada pregunta tiene hover effects y feedback visual
```

Paso 4: Envío del Formulario

```
1. Usuario hace clic en "Envoyer" (Enviar)
2. Sistema responde:
   - Efecto de confeti con partículas
   - Animación de sparkles
   - Redirección con mensaje de éxito
3. Datos se guardan en la base de datos
```

🛣️ RUTAS Y CONTROLADORES

Rutas Definidas en `routes/web.php`

1. Formulario de Registro (PÚBLICO)

```php
Route::get('/moods/{employee_id}/create', [MoodEmotionController::class,
'create'])
    ->name('moods.create')
    ->middleware('setlocale');
```

- **URL:** `/moods/E001/create`
- **Método:** GET
- **Controlador:** `MoodEmotionController@create`
- **Vista:** `resources/views/moods/create.blade.php`
- **Acceso:** Público (sin autenticación)
- **Middleware:** `setlocale` (para idiomas)

2. Guardar Emoción (PÚBLICO)

```php
Route::post('/moods/{employee_id}', [MoodEmotionController::class, 'store'])
    ->name('moods.store')
    ->middleware('setlocale');
```

- **URL:** `/moods/E001`
- **Método:** POST
- **Controlador:** `MoodEmotionController@store`
- **Acceso:** Público (sin autenticación)
- **Función:** Guarda datos en base de datos

3. Historial de Emociones (AUTENTICADO)

```
Route::get('/emotion/history', [MoodEmotionController::class, 'index'])
    ->name('emotion.index');
```

- **URL:** `/emotion/history`
- **Método:** GET
- **Controlador:** `MoodEmotionController@index`
- **Acceso:** Requiere autenticación
- **Función:** Muestra historial de emociones

Controlador: `MoodEmotionController`

Método `create()`

```
public function create($employee_id)
{
    // Prepara traducciones para JavaScript
    $translations = [
        'questions' => [
            'heureux' => [
                'q1' => __('moods.questions.heureux.q1'),
                'q2' => __('moods.questions.heureux.q2'),
                'q3' => __('moods.questions.heureux.q3'),
            ],
            // ... más emociones
        ],
        'motivational_messages' => [
            'heureux' => [
                'title' => __('moods.motivational_messages.heureux.title'),
                'message' => __('moods.motivational_messages.heureux.message'),
            ],
            // ... más mensajes
        ]
    ];

    return view('moods.create', compact('employee_id', 'translations'));
}
```

Método `store()`

```php
public function store(Request $request, $employee_id)
{
    // Validación de datos
    $data = $request->validate([
        'emotion' => 'required|in:heureux,neutre,frustre,tendu,calme',
        'answer_1' => 'required|in:0,1',
        'answer_2' => 'required|in:0,1',
        'answer_3' => 'required|in:0,1',
        'diary_text' => 'nullable|string',
    ]);

    // Guardar en base de datos
    MoodEmotion::create([
        'employee_id' => $employee_id,
        'emotion' => $data['emotion'],
        'answer_1' => $data['answer_1'],
        'answer_2' => $data['answer_2'],
        'answer_3' => $data['answer_3'],
        'diary_text' => '',
    ]);

    // Redirección con mensaje de éxito
    return redirect()
        ->route('moods.create', ['employee_id' => $employee_id])
        ->with('success', "Encouragement {$employee_id} enregistré avec succès");
}
```

📊 ESTRUCTURA DE DATOS

Modelo: `MoodEmotion`

```php
class MoodEmotion extends Model
{
    protected $table = 'mood_emotions';

    protected $fillable = [
        'employee_id',
        'emotion',
        'answer_1',
        'answer_2',
        'answer_3',
        'diary_text'
    ];
}
```

Tabla: `mood_emotions`

```
CREATE TABLE mood_emotions (
    id BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    employee_id VARCHAR(255) NOT NULL,
    emotion ENUM('heureux', 'neutre', 'frustre', 'tendu', 'calme') NOT NULL,
    answer_1 BOOLEAN NOT NULL,
    answer_2 BOOLEAN NOT NULL,
    answer_3 BOOLEAN NOT NULL,
    diary_text TEXT,
    created_at TIMESTAMP NULL,
    updated_at TIMESTAMP NULL
);
```

Ejemplo de Datos Guardados

```
// Cuando un empleado envía el formulario:
MoodEmotion::create([
    'employee_id' => 'E001',
    'emotion' => 'heureux',        // Feliz
    'answer_1' => 1,               // Sí a pregunta 1
    'answer_2' => 0,               // No a pregunta 2
    'answer_3' => 1,               // Sí a pregunta 3
    'diary_text' => '',            // Campo de texto (vacío)
]);
```

COMPONENTES FRONTEND

Vista Principal: `create.blade.php`

Estructura HTML

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>{{ __('moods.title') }}</title>
    @vite(['resources/css/app.css'])
</head>
<body class="bg-gray-50 min-h-screen font-sans p-6">
    @include('components.language-selector')
    <div class="max-w-2xl md:max-w-6xl lg:max-w-4xl mx-auto">
        <!-- Título -->
        <h1 class="text-2xl md:text-4xl lg:text-3xl font-bold text-center mb-6
md:mb-8">
            {{ __('moods.header', ['employee_id' => $employee_id]) }}
        </h1>
```

```
        <!-- Saludo personalizado -->
        <div class="text-center mb-4 md:mb-6">
            <p class="text-base md:text-xl lg:text-lg text-gray-600"
id="greeting">
                <!-- Saludo según hora del día -->
            </p>
        </div>

        <!-- Indicador de progreso -->
        <div class="flex justify-center mb-6 md:mb-8">
            <!-- Pasos 1 y 2 -->
        </div>

        <!-- Formulario -->
        <form action="{{ route('moods.store', ['employee_id' => $employee_id]) }}"
method="POST">
            @csrf
            <input type="hidden" name="employee_id" value="{{ $employee_id }}">

            <!-- Selección de emoción -->
            <div class="flex justify-center items-center space-x-6 md:space-x-12
lg:space-x-8 mb-8">
                <!-- 5 emojis con labels -->
            </div>

            <!-- Preguntas (ocultas inicialmente) -->
            <div id="questions" class="hidden space-y-6">
                <!-- 3 preguntas con radio buttons -->
            </div>

            <!-- Botón de envío -->
            <button type="submit" class="w-full py-3 md:py-4 lg:py-4 bg-gradient-
to-r from-primary to-secondary text-white rounded-xl font-semibold text-base
md:text-lg lg:text-lg min-h-[48px] md:min-h-[56px] lg:min-h-[56px]">
                {{__('moods.submit')}}
            </button>
        </form>
    </div>

    <!-- JavaScript -->
    <script>
        window.translations = @json($translations);
    </script>
    <script src="{{ asset('js/mood-form.js') }}"></script>
</body>
</html>
```

JavaScript: `mood-form.js`

Funcionalidades Principales

```
document.addEventListener('DOMContentLoaded', () => {
    // 1. Selección de emoción
    emotionRadio.forEach(radio => {
        radio.addEventListener('change', () => {
            // Mostrar preguntas específicas
            // Aplicar colores según emoción
            // Mostrar mensaje motivacional
            // Actualizar progreso
        });
    });

    // 2. Navegación por teclado
    function handleEmotionKeydown(event, emotionValue) {
        // Navegación con flechas
        // Selección con Enter/Espacio
    }

    // 3. Efecto de confeti
    form.addEventListener('submit', function(e) {
        createConfetti();
    });
});
```

Estilos CSS: `app.css`

Animaciones Personalizadas

```
/* Animaciones para partículas */
@keyframes fall-circle {
    to {
        transform: translateY(100vh) rotate(720deg);
    }
}

@keyframes sparkle {
    0% { opacity: 0; transform: scale(0) rotate(0deg); }
    50% { opacity: 1; transform: scale(1.2) rotate(180deg); }
    100% { opacity: 0; transform: scale(0) rotate(360deg); }
}

/* Efectos de pulso */
@keyframes pulse-glow {
    0%, 100% { box-shadow: 0 0 5px rgba(48, 207, 208, 0.5); }
    50% { box-shadow: 0 0 20px rgba(48, 207, 208, 0.8); }
}
```

🔄 FLUJO DE DATOS

1. Entrada de Datos

```
Usuario → Formulario HTML → JavaScript → Validación → Envío POST
```

2. Procesamiento Backend

```
Request POST → MoodEmotionController@store → Validación → MoodEmotion::create →
Base de Datos
```

3. Respuesta al Usuario

```
Base de Datos → Controlador → Redirección → Vista → Mensaje de Éxito + Confeti
```

4. Flujo Completo

```
1. Usuario visita /moods/E001/create
   ↓
2. MoodEmotionController@create prepara datos
   ↓
3. Vista create.blade.php se renderiza
   ↓
4. Usuario interactúa con JavaScript
   ↓
5. Formulario se envía a /moods/E001 (POST)
   ↓
6. MoodEmotionController@store procesa datos
   ↓
7. Datos se guardan en mood_emotions
   ↓
8. Usuario es redirigido con mensaje de éxito
```

SEGURIDAD ACTUAL

Medidas de Seguridad Implementadas

1. Validación de Datos

```php
$data = $request->validate([
    'emotion' => 'required|in:heureux,neutre,frustre,tendu,calme',
    'answer_1' => 'required|in:0,1',
    'answer_2' => 'required|in:0,1',
```

```
    'answer_3' => 'required|in:0,1',
]);
```

2. Protección CSRF

```
@csrf  <!-- Token CSRF en formularios -->
```

3. Middleware de Idioma

```
->middleware('setlocale')  // Para manejo de idiomas
```

Limitaciones de Seguridad Actual

1. Sin Autenticación

- ☑ **Ventaja:** Acceso fácil para empleados
- ❌ **Desventaja:** Cualquiera puede registrar emociones

2. Sin Validación de Employee ID

- ❌ **Riesgo:** Usuario puede cambiar employee_id en URL
- ❌ **Riesgo:** Registros falsos con IDs inexistentes

3. Sin Rate Limiting

- ❌ **Riesgo:** Spam de registros
- ❌ **Riesgo:** Ataques de denegación de servicio

📈 MÉTRICAS ACTUALES

Datos que se Capturan

- **Empleado:** ID del empleado
- **Emoción:** 5 tipos posibles
- **Respuestas:** 3 preguntas binarias
- **Timestamp:** Fecha y hora del registro
- **Idioma:** Configuración del usuario

Datos que NO se Capturan

- **IP del usuario**
- **Dispositivo utilizado**
- **Tiempo de respuesta**
- **Intentos fallidos**

- **Sesión del usuario**

🎯 PRÓXIMOS PASOS

Dashboard (Próxima Implementación)

- **Ruta:** `/dashboard` (protegida)
- **Acceso:** Solo administradores
- **Funcionalidad:** Gráficos y estadísticas globales
- **Privacidad:** Sin datos individuales de empleados

Mejoras de Seguridad Sugeridas

- Implementar validación de employee_id
- Añadir rate limiting
- Logging de actividades
- Validación de dispositivos

Documento generado el: 07/08/2025
Versión del sistema: 1.0
Estado: Funcional ☑