GUÍA COMPLETA: SISTEMA MULTIIDIOMA EN LARAVEL

OBJETIVO

Implementar un sistema completo de cambio de idioma (Francés, Inglés, Español) en un proyecto Laravel con:

- Idioma por defecto configurable
- Selector visual de idioma
- Traducciones dinámicas en vistas y JavaScript
- Middleware para detectar preferencias de usuario

PASO 1: CONFIGURACIÓN INICIAL

1.1 Cambiar idioma por defecto

Archivo: `config/app.php`

```
'locale' => 'fr', // Cambiar de 'en' a 'fr' (o el idioma que prefieras)
```

1.2 Crear estructura de carpetas

```
lang/
├── en/
│   ├── app.php
│   └── moods.php
├── es/
│   ├── app.php
│   └── moods.php
└── fr/
    ├── app.php
    └── moods.php
```

PASO 2: CREAR ARCHIVOS DE TRADUCCIÓN

2.1 Estructura de archivo de traducción

Archivo: `lang/fr/moods.php` (ejemplo)

```
<?php

return [
```

```
    // Títulos y encabezados
    'title' => 'Enregistrer le Moral',
    'header' => 'Enregistrer le moral de l\'employé :employee_id',
    'how_do_you_feel' => 'Comment vous sentez-vous aujourd\'hui?',
    'answer_questions' => 'Répondez aux questions suivantes',

    // Emociones (formato: emoji|texto)
    'emotions' => [
        'heureux' => '😊|Heureux',
        'neutre' => '😐|Neutre',
        'frustre' => '😞|Frustré',
        'tendu' => '😣|Tendu',
        'calme' => '😌|Calme',
    ],

    // Preguntas dinámicas por emoción
    'questions' => [
        'heureux' => [
            'q1' => 'Vous êtes-vous senti motivé pour commencer votre journée?',
            'q2' => 'Avez-vous eu le sentiment que vos tâches avaient un objectif
clair?',
            'q3' => 'Avez-vous eu des interactions positives avec vos collègues?',
        ],
        // ... otras emociones
    ],

    // Botones
    'yes' => 'Oui',
    'no' => 'Non',
    'submit' => 'Envoyer',

    // Mensajes
    'success_message' => 'Moral enregistré avec succès pour l\'employé
:employee_id',
];
```

2.2 Crear archivos para cada idioma

Repetir la estructura para:

- `lang/en/moods.php` (traducciones en inglés)
- `lang/es/moods.php` (traducciones en español)

🔧 PASO 3: CREAR MIDDLEWARE

3.1 Crear archivo middleware

Archivo: `app/Http/Middleware/SetLocale.php`

```php
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Http\Request;
use Symfony\Component\HttpFoundation\Response;

class SetLocale
{
    /**
     * Handle an incoming request.
     */
    public function handle(Request $request, Closure $next): Response
    {
        // Verificar si hay parámetro 'lang' en la URL
        if ($request->has('lang')) {
            $locale = $request->get('lang');

            // Verificar que el idioma sea válido
            if (in_array($locale, ['en', 'es', 'fr'])) {
                // Cambiar el idioma de la aplicación
                app()->setLocale($locale);

                // Guardar la preferencia en la sesión
                session()->put('locale', $locale);
            }
        }
        // Si no hay parámetro en URL, usar la sesión
        elseif (session()->has('locale')) {
            app()->setLocale(session()->get('locale'));
        }

        return $next($request);
    }
}
```

3.2 Registrar middleware

Archivo: `bootstrap/app.php`

```php
->withMiddleware(function (Middleware $middleware): void {
    $middleware->alias([
        'setlocale' => \App\Http\Middleware\SetLocale::class,
    ]);
})
```

PASO 4: CONFIGURAR RUTAS

4.1 Aplicar middleware a rutas

Archivo: routes/web.php

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

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

🎛️ PASO 5: CREAR COMPONENTE SELECTOR

5.1 Crear componente

Archivo: resources/views/components/language-selector.blade.php

```
<div class="flex justify-end mb-4">
    <div class="flex space-x-2">
        <a href="{{ request()->fullUrlWithQuery(['lang' => 'fr']) }}"
           class="px-3 py-1 rounded {{ app()->getLocale() === 'fr' ? 'bg-blue-500
text-white' : 'bg-gray-200 text-gray-700' }} hover:bg-blue-600 transition">
            FR
        </a>
        <a href="{{ request()->fullUrlWithQuery(['lang' => 'en']) }}"
           class="px-3 py-1 rounded {{ app()->getLocale() === 'en' ? 'bg-blue-500
text-white' : 'bg-gray-200 text-gray-700' }} hover:bg-blue-600 transition">
            EN
        </a>
        <a href="{{ request()->fullUrlWithQuery(['lang' => 'es']) }}"
           class="px-3 py-1 rounded {{ app()->getLocale() === 'es' ? 'bg-blue-500
text-white' : 'bg-gray-200 text-gray-700' }} hover:bg-blue-600 transition">
            ES
        </a>
    </div>
</div>
```

5.2 Agregar a vista

Archivo: resources/views/moods/create.blade.php

```
<div class="container mx-auto px-4 py-8">
    <x-language-selector />
    <!-- Resto del contenido -->
</div>
```

🎮 PASO 6: MODIFICAR CONTROLADOR

6.1 Pasar traducciones a la vista

Archivo: app/Http/Controllers/MoodEmotionController.php

```
public function create($employee_id)
{
    // Crear arrays con todas las traducciones de preguntas
    $translations = [
        'questions' => [
            'heureux' => [
                'q1' => __('moods.questions.heureux.q1'),
                'q2' => __('moods.questions.heureux.q2'),
                'q3' => __('moods.questions.heureux.q3'),
            ],
            'neutre' => [
                'q1' => __('moods.questions.neutre.q1'),
                'q2' => __('moods.questions.neutre.q2'),
                'q3' => __('moods.questions.neutre.q3'),
            ],
            // ... otras emociones
        ]
    ];

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

🎯 PASO 7: MODIFICAR VISTA

7.1 Cambiar textos hardcodeados

Reemplazar todos los textos por traducciones:

Antes:

```
<title>Enregistrer Mood</title>
<h1>Enregistrer le moral de l'employé {{ $employee_id }}</h1>
<p>Comment te sens-tu aujourd'hui ?</p>
<button>Enviar</button>
```

Después:

```
<title>{{ __('moods.title') }}</title>
<h1>{{ __('moods.header', ['employee_id' => $employee_id]) }}</h1>
<p>{{ __('moods.how_do_you_feel') }}</p>
<button>{{ __('moods.submit') }}</button>
```

7.2 Pasar traducciones a JavaScript

```
{{-- Pasar traducciones al JavaScript --}}
<script>
    // Hacer traducciones disponibles globalmente
    window.translations = @json($translations);
</script>
<script src="{{ asset('js/mood-form.js') }}"></script>
```

⚡ PASO 8: MODIFICAR JAVASCRIPT

8.1 Usar traducciones dinámicas

Archivo: `public/js/mood-form.js`

```
document.addEventListener('DOMContentLoaded', () => {
    const emotionRadio = document.querySelectorAll('input[name="emotion"]');
    const questionsDiv = document.getElementById('questions');

    const q1 = document.getElementById('q1');
    const q2 = document.getElementById('q2');
    const q3 = document.getElementById('q3');

    // ☑ Usar las traducciones del servidor
    const questionsParEmotion = window.translations.questions;

    emotionRadio.forEach(radio => {
        radio.addEventListener('change', () => {
            const emotion = radio.value;
            const questions = questionsParEmotion[emotion];

            q1.textContent = questions.q1;
            q2.textContent = questions.q2;
            q3.textContent = questions.q3;

            questionsDiv.classList.remove('hidden');
            questionsDiv.scrollIntoView({ behavior: 'smooth' });
        });
    });
});
```

🔧 FUNCIONES LARAVEL CLAVE

__() Helper

```
// Traducción simple
__('moods.title')

// Traducción con parámetros
__('moods.header', ['employee_id' => $employee_id])

// Traducción con fallback
__('moods.title', [], 'en')
```

app()->getLocale()

```
// Obtener idioma actual
$currentLocale = app()->getLocale();

// Cambiar idioma
app()->setLocale('fr');
```

session()->put() y session()->get()

```
// Guardar en sesión
session()->put('locale', 'fr');

// Obtener de sesión
$locale = session()->get('locale');
```

🚨 PROBLEMAS COMUNES Y SOLUCIONES

1. Emojis corruptos

Problema: Los emojis se ven como ``

Solución: Asegurar que el archivo se guarde en UTF-8

2. Middleware no funciona

Problema: No se detecta el cambio de idioma

Solución: Verificar que esté registrado en `bootstrap/app.php`

3. Traducciones no aparecen

Problema: Se muestran las claves en lugar del texto

Solución: Verificar que existan en los archivos de traducción

4. JavaScript no recibe traducciones

Problema: `window.translations` es undefined

Solución: Verificar que se pase desde el controlador y se use `@json()`

☑ CHECKLIST FINAL

- ☐ Idioma por defecto configurado en `config/app.php`
- ☐ Archivos de traducción creados para todos los idiomas
- ☐ Middleware `SetLocale` creado y registrado
- ☐ Rutas con middleware `setlocale` aplicado
- ☐ Componente selector de idioma creado
- ☐ Vistas usando `__()` helper
- ☐ Controlador pasando traducciones a JavaScript
- ☐ JavaScript usando `window.translations`
- ☐ Pruebas realizadas en todos los idiomas

💡 CONSEJOS ADICIONALES

- **Usar claves descriptivas** en los archivos de traducción
- **Mantener consistencia** en la estructura de traducciones
- **Probar en todos los idiomas** antes de finalizar
- **Usar parámetros** para textos dinámicos
- **Documentar** las claves de traducción para el equipo

🎉 ¡SISTEMA MULTIIDIOMA COMPLETADO!

Ahora tu aplicación Laravel tiene un sistema completo de cambio de idioma que permite a los usuarios cambiar entre Francés, Inglés y Español de forma dinámica, con persistencia de preferencias y traducciones tanto en el servidor como en el cliente.

📄 EXPORTAR A PDF

Opción 1: Con Pandoc (si está instalado)

```
pandoc guia_multiidioma_laravel.md -o guia_multiidioma_laravel.pdf
```

Opción 2: Con herramientas online

- **StackEdit** (stackedit.io)

- **Dillinger** (dillinger.io)
- **Markdown to PDF** (markdowntopdf.com)

Opción 3: Con editores de código

- **VS Code** con extensión "Markdown PDF"
- **Typora** (editor Markdown con exportación a PDF)
- **Obsidian** (con plugin de exportación)

Opción 4: Con navegador

1. Abrir el archivo .md en un editor online
2. Usar la función de exportar a PDF
3. Descargar el archivo PDF generado