

DOCUMENTACIÓN COMPLETA: PANEL DE ADMINISTRACIÓN

GBN Emotions - Sistema de Gestión de Emociones

Fecha: 8 de Agosto de 2025

Versión: 1.0

Autor: Asistente IA + Usuario

Proyecto: GBN Emotions - Panel de Administración

ÍNDICE

1. [Resumen Ejecutivo](#)
 2. [Arquitectura del Sistema](#)
 3. [Controlador AdminController](#)
 4. [Vista del Panel de Administración](#)
 5. [Sistema de Autenticación](#)
 6. [Rutas y Middleware](#)
 7. [Base de Datos](#)
 8. [Problemas Resueltos](#)
 9. [Comandos Utilizados](#)
 10. [Estructura de Archivos](#)
-

RESUMEN EJECUTIVO




Objetivo del Panel:

Crear un panel de administración exclusivo para managers que permita visualizar estadísticas agregadas y anónimas de las emociones de los empleados, respetando la privacidad individual.

Funcionalidades Implementadas:

- ☒ **Estadísticas Generales** - Porcentajes de emociones positivas, negativas y neutras
- ☒ **Gráfico de Tendencia** - Visualización de la distribución emocional
- ☒ **Emociones Más Frecuentes** - Tabla con conteos y porcentajes por emoción
- ☒ **Sistema de Autenticación** - Protección del panel con login
- ☒ **Diseño Responsive** - Interfaz moderna con Tailwind CSS

Principios de Privacidad:

-  **Datos Agregados** - Solo estadísticas generales, no datos individuales
 -  **Anonimización** - No se muestran empleados específicos
 -  **Protección de Derechos** - Cumplimiento con derechos laborales
-

ARQUITECTURA DEL SISTEMA

Flujo de Datos:

Usuario → Login → Middleware Auth → AdminController → Vista → Estadísticas

Componentes Principales:

1. **AdminController** - Lógica de negocio y consultas
2. **Vista admin/dashboard.blade.php** - Interfaz de usuario
3. **Rutas protegidas** - Control de acceso
4. **Sistema de autenticación** - Seguridad
5. **Base de datos** - Almacenamiento de datos

CONTROLADOR ADMINCONTROLLER

Ubicación: `app/Http/Controllers/AdminController.php`

Estructura Completa del Controlador:

```
<?php

namespace App\Http\Controllers;

use App\Models\MoodEmotion;
use App\Models\Department;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class AdminController extends Controller
{
    /**
     * Panel principal de administración
     * Muestra estadísticas generales y emociones más frecuentes
     */
    public function index()
    {
        // =====
        // CONFIGURACIÓN DE CATEGORÍAS DE EMOCIONES
        // =====
        $emotions = ['heureux', 'neutre', 'frustre', 'tendu', 'calme'];
        $positiveEmotions = ['heureux', 'calme'];
        $negativeEmotions = ['frustre', 'tendu'];
        $neutralEmotions = ['neutre'];

        // =====
        // CÁLCULO DE TENDENCIA GENERAL
        // =====
    }
}
```

```

        // Contar registros por categoría de emoción
        $positiveCount = MoodEmotion::whereIn('emotion', $positiveEmotions)-
>count();
        $negativeCount = MoodEmotion::whereIn('emotion', $negativeEmotions)-
>count();
        $neutralCount = MoodEmotion::whereIn('emotion', $neutralEmotions)-
>count();

        // Calcular porcentajes de cada categoría
        $totalCount = $positiveCount + $negativeCount + $neutralCount;
        $percentagePositive = ($positiveCount / $totalCount) * 100;
        $percentageNegative = ($negativeCount / $totalCount) * 100;
        $percentageNeutral = ($neutralCount / $totalCount) * 100;

        // =====
        // EMOCIONES MÁS FRECUENTES
        // =====

        // Obtener emociones con conteo y porcentaje
        $mostFrequentEmotions = MoodEmotion::select(
            'emotion',
            DB::raw('COUNT(*) as count'),
            DB::raw('(COUNT(*) * 100.0 / (SELECT COUNT(*) FROM mood_emotions)) as
percentage')
        )
        ->groupBy('emotion')
        ->orderBy('count', 'desc')
        ->get();

        // =====
        // RETORNAR DATOS A LA VISTA
        // =====

        return view('admin.dashboard', [
            // Datos de tendencia general
            'totalRecords' => $totalCount,
            'percentagePositive' => $percentagePositive,
            'percentageNegative' => $percentageNegative,
            'percentageNeutral' => $percentageNeutral,

            // Datos de emociones más frecuentes
            'mostFrequentEmotions' => $mostFrequentEmotions,

            // Datos adicionales para estadísticas detalladas
            'positiveCount' => $positiveCount,
            'negativeCount' => $negativeCount,
            'neutralCount' => $neutralCount
        ]);
    }
}

```

Explicación Línea por Línea:

1. Imports y Namespace:

```
namespace App\Http\Controllers;  
use App\Models\MoodEmotion;  
use App\Models\Department;  
use Illuminate\Http\Request;  
use Illuminate\Support\Facades\DB;
```

- **MoodEmotion** - Modelo para acceder a los registros de emociones
- **Department** - Modelo para futuras funcionalidades por departamento
- **Request** - Para manejar peticiones HTTP
- **DB** - Facade para consultas SQL personalizadas (necesario para `DB::raw()`)

2. Configuración de Categorías:

```
$emotions = ['heureux', 'neutre', 'frustre', 'tendu', 'calme'];  
$positiveEmotions = ['heureux', 'calme'];  
$negativeEmotions = ['frustre', 'tendu'];  
$neutralEmotions = ['neutre'];
```

- **\$emotions** - Array con todas las emociones disponibles
- **\$positiveEmotions** - Emociones clasificadas como positivas
- **\$negativeEmotions** - Emociones clasificadas como negativas
- **\$neutralEmotions** - Emociones clasificadas como neutras

3. Cálculo de Conteos:

```
$positiveCount = MoodEmotion::whereIn('emotion', $positiveEmotions)->count();  
$negativeCount = MoodEmotion::whereIn('emotion', $negativeEmotions)->count();  
$neutralCount = MoodEmotion::whereIn('emotion', $neutralEmotions)->count();
```

- **whereIn('emotion', \$array)** - Filtra registros donde la emoción esté en el array
- **count()** - Cuenta el número de registros que cumplen la condición
- **Resultado:** Número total de registros por cada categoría

4. Cálculo de Porcentajes:

```
$totalCount = $positiveCount + $negativeCount + $neutralCount;  
$percentagePositive = ($positiveCount / $totalCount) * 100;  
$percentageNegative = ($negativeCount / $totalCount) * 100;  
$percentageNeutral = ($neutralCount / $totalCount) * 100;
```

- **\$totalCount** - Suma de todos los registros válidos
- **Fórmula:** $(\text{conteo_categoría} / \text{total}) * 100$
- **Resultado:** Porcentaje de cada categoría sobre el total

5. Consulta de Emociones Más Frecuentes:

```
$mostFrequentEmotions = MoodEmotion::select(
    'emotion',
    DB::raw('COUNT(*) as count'),
    DB::raw('(COUNT(*) * 100.0 / (SELECT COUNT(*) FROM mood_emotions)) as
percentage')
)
->groupBy('emotion')
->orderBy('count', 'desc')
->get();
```

Explicación detallada:

- **select('emotion')** - Selecciona la columna de emoción
- **DB::raw('COUNT(*) as count')** - Cuenta registros por emoción, alias 'count'
- **DB::raw('(COUNT(*) * 100.0 / (SELECT COUNT(*) FROM mood_emotions)) as percentage')**
 - Calcula porcentaje en la consulta SQL
 - **COUNT(*)** - Conteo de registros de esta emoción
 - *** 100.0** - Multiplica por 100 para porcentaje
 - **/ (SELECT COUNT(*) FROM mood_emotions)** - Divide por el total de registros
 - **as percentage** - Alias del resultado
- **groupBy('emotion')** - Agrupa resultados por tipo de emoción
- **orderBy('count', 'desc')** - Ordena por conteo descendente (más frecuente primero)
- **get()** - Ejecuta la consulta y retorna la colección

6. Retorno de Datos:

```
return view('admin.dashboard', [
    'totalRecords' => $totalCount,
    'percentagePositive' => $percentagePositive,
    'percentageNegative' => $percentageNegative,
    'percentageNeutral' => $percentageNeutral,
    'mostFrequentEmotions' => $mostFrequentEmotions,
    'positiveCount' => $positiveCount,
    'negativeCount' => $negativeCount,
    'neutralCount' => $neutralCount
]);
```

- **view('admin.dashboard')** - Renderiza la vista ubicada en `resources/views/admin/dashboard.blade.php`

- **Array asociativo** - Pasa variables a la vista
- **Variables disponibles en la vista:**
 - `$totalRecords` - Total de registros
 - `$percentagePositive/Negative/Neutral` - Porcentajes por categoría
 - `$mostFrequentEmotions` - Colección con emociones y sus estadísticas
 - `$positiveCount/NegativeCount/NeutralCount` - Conteos absolutos

VISTA DEL PANEL DE ADMINISTRACIÓN

Ubicación: `resources/views/admin/dashboard.blade.php`

Estructura de la Vista:

1. Layout y Header:

```
<x-app-layout>
  <x-slot name="header">
    <h2 class="font-semibold text-xl text-gray-800 leading-tight">
      {{ __('Panel de Administración') }}
    </h2>
  </x-slot>
```

- `<x-app-layout>` - Componente Blade que usa el layout `app.blade.php`
- `<x-slot name="header">` - Define el contenido del header
- `{{ __('Panel de Administración') }}` - Función de traducción (i18n)

2. Estadísticas Generales:

```
<div class="grid grid-cols-1 md:grid-cols-4 gap-4">
  <!-- Total de registros -->
  <div class="bg-blue-50 p-4 rounded-lg border border-blue-200">
    <div class="text-2xl font-bold text-blue-600">{{
number_format($totalRecords) }}</div>
    <div class="text-sm text-gray-600">Total de registros</div>
  </div>

  <!-- Emociones positivas -->
  <div class="bg-green-50 p-4 rounded-lg border border-green-200">
    <div class="text-2xl font-bold text-green-600">{{
number_format($percentagePositive, 1) }}%</div>
    <div class="text-sm text-gray-600">Emociones positivas</div>
    <div class="text-xs text-gray-500">{{ $positiveCount }} registros</div>
  </div>

  <!-- Emociones negativas -->
  <div class="bg-red-50 p-4 rounded-lg border border-red-200">
    <div class="text-2xl font-bold text-red-600">{{
```

```

number_format($percentageNegative, 1) }}%</div>
    <div class="text-sm text-gray-600">Emociones negativas</div>
    <div class="text-xs text-gray-500">{{ $negativeCount }} registros</div>
</div>

<!-- Emociones neutras -->
<div class="bg-gray-50 p-4 rounded-lg border border-gray-200">
    <div class="text-2xl font-bold text-gray-600">{{
number_format($percentageNeutral, 1) }}%</div>
    <div class="text-sm text-gray-600">Emociones neutras</div>
    <div class="text-xs text-gray-500">{{ $neutralCount }} registros</div>
</div>
</div>

```

Explicación de clases Tailwind:

- **grid grid-cols-1 md:grid-cols-4** - Grid responsive: 1 columna en móvil, 4 en desktop
- **bg-blue-50** - Fondo azul claro
- **text-2xl font-bold** - Texto grande y negrita
- **number_format(\$variable, 1)** - Formatea números con 1 decimal

3. Gráfico de Tendencia:

```

<div class="w-full bg-gray-200 rounded-full h-8">
    <div class="bg-green-500 h-8 rounded-l-full" style="width: {{
$percentagePositive }}%"></div>
    <div class="bg-red-500 h-8" style="width: {{ $percentageNegative }}%; margin-
left: -{{ $percentagePositive }}%"></div>
    <div class="bg-gray-500 h-8 rounded-r-full" style="width: {{
$percentageNeutral }}%; margin-left: -{{ $percentageNegative }}%"></div>
</div>

```

Lógica del gráfico:

- **Barra base gris** - Fondo del gráfico
- **Barra verde** - Porcentaje de emociones positivas
- **Barra roja** - Porcentaje de emociones negativas (con margen negativo para solaparse)
- **Barra gris** - Porcentaje de emociones neutras (con margen negativo para solaparse)

4. Tabla de Emociones Frecuentes:

```

@foreach($mostFrequentEmotions as $emotion)
<tr class="hover:bg-gray-50">
    <td class="px-6 py-4 whitespace-nowrap">
        <div class="flex items-center">
            <div class="text-sm font-medium text-gray-900">
                @switch($emotion->emotion)
                @case('heureux')

```

```

        😊 Heureux (Feliz)
        @break
        @case('calme')
        😌 Calme (Tranquilo)
        @break
        @case('neutre')
        😐 Neutre (Neutral)
        @break
        @case('frustre')
        😡 Frustré (Frustrado)
        @break
        @case('tendu')
        😬 Tendu (Tenso)
        @break
        @default
        {{ $emotion->emotion }}
    @endswitch
</div>
</div>
</td>
<td class="px-6 py-4 whitespace-nowrap">
    <div class="text-sm text-gray-900">{{ number_format($emotion->count) }}
</div>
</td>
<td class="px-6 py-4 whitespace-nowrap">
    <div class="text-sm text-gray-900">{{ number_format($emotion->percentage,
1) }}%</div>
</td>
<td class="px-6 py-4 whitespace-nowrap">
    <div class="w-32 bg-gray-200 rounded-full h-2">
        <div class="bg-blue-500 h-2 rounded-full" style="width: {{ $emotion-
>percentage }}%"></div>
    </div>
</td>
</tr>
</div>
</div>

```

Explicación del switch:

- `@switch($emotion->emotion)` - Evalúa el valor de la emoción
- `@case('heureux')` - Si la emoción es 'heureux'
- 😊 **Heureux (Feliz)** - Muestra emoji + nombre en francés + traducción
- `@break` - Termina la evaluación
- `@default` - Si no coincide ningún caso

🔒 SISTEMA DE AUTENTICACIÓN

Usuario Administrador Creado:

Credenciales:

- **Nombre:** Admin
- **Contraseña:** admin123

Proceso de Creación:

1. Creación del Seeder:

```
php artisan make:seeder AdminUserSeeder
```

2. Contenido del Seeder (`database/seeder/AdminUserSeeder.php`):

```
<?php

namespace Database\Seeders;

use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;
use App\Models\User;
use Illuminate\Support\Facades\Hash;

class AdminUserSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        // Crear usuario administrador
        User::create([
            'name' => 'Admin',
            'password' => Hash::make('admin123'),
        ]);
    }
}
```

3. Ejecución del Seeder:

```
php artisan db:seed --class=AdminUserSeeder
```

Explicación del Hashing:

```
Hash::make('admin123')
```

- **Función:** Convierte la contraseña en texto plano a un hash seguro

- **Resultado:** \$2y\$12\$jS/7MtEyFUEQSS8apj7jY.IRBYQYv0IXvEw8.7/pJMSrRX2BPwLm6
- **Características:**
 - **Irreversible** - No se puede desenscriptar
 - **Único** - Cada ejecución genera un hash diferente
 - **Seguro** - Usa algoritmo bcrypt con salt



RUTAS Y MIDDLEWARE

Ubicación: routes/web.php

Ruta del Panel de Administración:

```
// Panel de Administración (solo para administradores)
Route::get('/admin', [AdminController::class, 'index'])
    ->middleware(['auth', 'verified'])
    ->name('admin.dashboard');
```

Explicación de la Ruta:

- **Route::get('/admin')** - Ruta GET que responde a /admin
- **[AdminController::class, 'index']** - Controlador y método a ejecutar
- **->middleware(['auth', 'verified'])** - Middleware de autenticación
 - **auth** - Verifica que el usuario esté autenticado
 - **verified** - Verifica que el email esté verificado (opcional)
- **->name('admin.dashboard')** - Nombre de la ruta para generar URLs

Import del Controlador:

```
use App\Http\Controllers\AdminController;
```



BASE DE DATOS

Tabla Users (Estructura):

```
CREATE TABLE users (
    id INTEGER PRIMARY KEY,
    name VARCHAR NOT NULL,
    password VARCHAR NOT NULL,
    remember_token VARCHAR NULL,
    created_at TIMESTAMP NULL,
    updated_at TIMESTAMP NULL
);
```

Tabla Mood_Emotions (Estructura):

```
CREATE TABLE mood_emotions (  
  id INTEGER PRIMARY KEY,  
  employee_id VARCHAR NOT NULL,  
  emotion VARCHAR NOT NULL,  
  answer_1 INTEGER NOT NULL,  
  answer_2 INTEGER NOT NULL,  
  answer_3 INTEGER NOT NULL,  
  diary_text TEXT NULL,  
  department_id INTEGER NULL,  
  created_at TIMESTAMP NULL,  
  updated_at TIMESTAMP NULL,  
  FOREIGN KEY (department_id) REFERENCES departments(id)  
);
```

Relaciones:

- **Users ↔ MoodEmotions** - No hay relación directa (privacidad)
- **Departments ↔ MoodEmotions** - One-to-Many (un departamento tiene muchas emociones)

⚠ PROBLEMAS RESUELTOS

1. Error: "Undefined variable \$slot"

Problema: El layout `app.blade.php` usaba `@yield` en lugar de `{{ $slot }}`

Solución: Actualizar el layout para usar componentes Blade:

```
// ANTES (no funcionaba):  
@yield('header')  
@yield('content')  
  
// DESPUÉS (funciona):  
{{ $header ?? '' }}  
{{ $slot }}
```

2. Error: "Route [login] not defined"

Problema: Faltaban las rutas de autenticación

Solución: Verificar que `require __DIR__ . '/auth.php';` esté en `routes/web.php`

3. Error: "table users has no column named email"

Problema: Intenté crear usuario con campo `email` que no existe

Solución: Usar solo los campos disponibles: `name` y `password`

4. Error: "Undefined constant App\Models\Department"

Problema: Tinker no reconocía los modelos

Solución: Usar `composer dump-autoload` para regenerar autoload



COMANDOS UTILIZADOS

Creación de Archivos:

```
# Crear controlador
php artisan make:controller AdminController

# Crear seeder
php artisan make:seeder AdminUserSeeder

# Crear directorio de vistas
mkdir -p resources/views/admin
```

Base de Datos:

```
# Ejecutar seeder específico
php artisan db:seed --class=AdminUserSeeder

# Regenerar autoload
composer dump-autoload

# Ver rutas
php artisan route:list | findstr admin
```

Servidor:

```
# Iniciar servidor de desarrollo
php artisan serve
```

Tinker (para debugging):

```
# Verificar usuarios
php artisan tinker --execute="DB::table('users')->get();"

# Verificar emociones
php artisan tinker --execute="DB::table('mood_emotions')->count();"
```

ESTRUCTURA DE ARCHIVOS

Archivos Creados/Modificados:

```
gbnEmotions/
├── app/
│   ├── Http/
│   │   └── Controllers/
│   │       └── AdminController.php          ← NUEVO
│   ├── database/
│   │   ├── seeders/
│   │   │   └── AdminUserSeeder.php        ← NUEVO
│   ├── resources/
│   │   ├── views/
│   │   │   ├── admin/
│   │   │   │   ├── dashboard.blade.php    ← NUEVO
│   │   │   │   └── layouts/
│   │   │   │       └── app.blade.php       ← MODIFICADO
│   ├── routes/
│   │   └── web.php                        ← MODIFICADO
└── DOCUMENTACION_PANEL_ADMIN_COMPLETO_2025_08_08.md ← NUEVO
```

Relaciones entre Archivos:

Flujo de Ejecución:



1. **routes/web.php** → Define la ruta `/admin`
2. **AdminController@index** → Procesa la lógica
3. **admin/dashboard.blade.php** → Renderiza la vista
4. **layouts/app.blade.php** → Proporciona el layout base

Dependencias:

- **AdminController** → **MoodEmotion Model** → **Base de datos**
- **Vista dashboard** → **Layout app** → **Tailwind CSS**
- **Rutas** → **Middleware auth** → **Sistema de autenticación**

PRÓXIMOS PASOS RECOMENDADOS

Funcionalidades a Implementar:

1.  **Gráficos por Departamento**
 - Estadísticas diferenciadas por departamento
 - Comparativas entre departamentos
2.  **Filtros por Fecha**
 - Selección de rangos de fechas

- Análisis temporal de tendencias

3. 📄 Exportación de Datos

- Exportar a Excel/CSV
- Generar reportes PDF

4. 🔔 Sistema de Alertas

- Alertas cuando las emociones negativas superen umbrales
- Notificaciones automáticas

5. 👤 Gestión de Usuarios Admin

- Múltiples administradores
- Roles y permisos

Mejoras Técnicas:

1. 🔑 Middleware de Autorización

- Distinguir entre empleados y administradores
- Permisos granulares

2. 📱 Optimización Mobile

- Mejorar experiencia en dispositivos móviles
- Gráficos interactivos

3. ⚡ Performance

- Caché de consultas pesadas
- Paginación para grandes volúmenes

📝 NOTAS TÉCNICAS

Consideraciones de Seguridad:

- ☒ **Autenticación requerida** para acceder al panel
- ☒ **Datos agregados** para proteger privacidad
- ☒ **Contraseñas hasheadas** con bcrypt
- ☒ **Middleware de verificación** de email

Consideraciones de Performance:

- ☒ **Consultas optimizadas** con índices apropiados
- ☒ **Cálculos en base de datos** para reducir procesamiento
- ☒ **Vistas responsive** para diferentes dispositivos

Consideraciones de UX:

- ☒ **Interfaz intuitiva** con colores significativos

- ☒ **Información clara** sobre privacidad
 - ☒ **Diseño moderno** con Tailwind CSS
 - ☒ **Feedback visual** con gráficos y estadísticas
-

ENLACES ÚTILES

Documentación Laravel:

- [Eloquent ORM](#)
- [Blade Templates](#)
- [Authentication](#)
- [Database Seeding](#)

Recursos Externos:

- [Tailwind CSS](#)
 - [Laravel Breeze](#)
-

SOPORTE Y MANTENIMIENTO

Para Modificaciones:

1. **Backup de base de datos** antes de cambios
2. **Testing en entorno de desarrollo**
3. **Documentación de cambios**
4. **Versionado con Git**

Para Troubleshooting:

1. **Revisar logs** en `storage/logs/laravel.log`
 2. **Verificar rutas** con `php artisan route:list`
 3. **Comprobar base de datos** con Tinker
 4. **Limpiar caché** con `php artisan cache:clear`
-

Documentación creada el 8 de Agosto de 2025

Última actualización: 8 de Agosto de 2025

Estado: Completada y funcional ☒