1v0 Ship v1.0 Faster

Mastering Laravel Notifications: Mail, SMS, and Slack

Laravel Notifications provide a clean and powerful way to send alerts to users across multiple channels such as **email**, **SMS**, and **Slack**. Whether you're building sign-up confirmations, payment updates, or system alerts, Laravel 12 makes it straightforward. In this guide, we'll cover how to configure providers in `.env`, create reusable notification classes, trigger them from controllers and events, display them in your application's UI, and test them effectively.

Configure Notification Channels in Laravel 12 (.env setup)

Laravel supports multiple notification channels. You configure them in the `.env` file. Here are some examples:

```
# Default Mail driver (SMTP)
MAIL_MAILER=smtp
MAIL_HOST=smtp.mailtrap.io
MAIL_PORT=587
MAIL_USERNAME=your_smtp_username
MAIL_PASSWORD=your_smtp_password
MAIL_ENCRYPTION=tls
MAIL_FROM_ADDRESS=no-reply@example.com
MAIL_FROM_NAME="My App"

# Mailgun
MAIL_MAILER=mailgun
MAILGUN_DOMAIN=your-domain.com
MAILGUN_SECRET=your-mailgun-key
MAILGUN_ENDPOINT=api.mailgun.net
```

```
# Postmark
MAIL_MAILER=postmark
POSTMARK_TOKEN=your-postmark-server-token

# Slack Webhook
SLACK_WEBHOOK_URL=https://hooks.slack.com/services/xxxx/yyyy/zzzz

# Nexmo (Vonage)
NEXMO_KEY=your-nexmo-key
NEXMO_SECRET=your-nexmo-secret
NEXMO_SMS_FROM=MyApp

# Twilio
TWILIO_SID=your-twilio-sid
TWILIO_TOKEN=your-twilio-token
TWILIO_FROM=+1234567890Code language: Bash (bash)
```

These environment variables tell Laravel which service to use for each channel. For example, `MAIL_MAILER=mailgun` will route all email notifications through Mailgun, and `SLACK_WEBHOOK_URL` allows Slack messages to be posted to your workspace.

Send Laravel Notifications via Mail (SMTP)

First, create a notification class with Artisan:

```
php artisan make:notification UserRegisteredNotificationCode language: Bash (bash)
```

This generates a file under `app/Notifications`. Inside, define the email content:

```
namespace App\Notifications;

use Illuminate\Bus\Queueable;
```

```
use Illuminate\Notifications\Notification;
use Illuminate\Notifications\Messages\MailMessage;

class UserRegisteredNotification extends Notification
{
    use Queueable;

    public function via(object $notifiable): array
    {
        return ['mail'];
    }

    public function toMail(object $notifiable): MailMessage
    {
        return (new MailMessage)
            ->subject('Welcome to My App')
            ->greeting('Hello ' . $notifiable->name)
            ->line('Thanks for joining our platform!')
            ->action('Visit Dashboard', url('/dashboard'))
            ->line('We’re excited to have you onboard.');
    }
}Code language: PHP (php)
```

`via()` defines the channel, while `toMail()` builds the email with subject, body, and a call-to-action button.

Send Laravel Notifications via SMS (Nexmo or Twilio)

Once your Nexmo or Twilio credentials are in `.env`, extend your notification to also deliver SMS:

```
use Illuminate\Notifications\Messages\NexmoMessage;
```

```
public function via(object $notifiable): array
{
    return ['mail', 'nexmo'];
}

public function toNexmo(object $notifiable)
{
    return (new NexmoMessage)
        ->content('Hi ' . $notifiable->name . ', welcome to My App!');
}Code language: PHP (php)
```

This ensures both an email and an SMS are sent when you call `notify()`. For Twilio, use a Twilio channel package with a `toTwilio()` method.

Send Laravel Notifications to Slack

After setting your `SLACK_WEBHOOK_URL`, you can notify a Slack channel:

```
use Illuminate\Notifications\Messages\SlackMessage;

public function via(object $notifiable): array
{
    return ['slack'];
}

public function toSlack(object $notifiable)
{
    return (new SlackMessage)
        ->success()
        ->content('□ New user registered: ' . $notifiable->email);
}Code language: PHP (php)
```

Whenever `$user->notify()` is called, a message will also appear in your Slack channel.

Trigger Notifications from a Controller

Notify a user immediately after registration inside your controller:

```
$user->notify(new UserRegisteredNotification());Code language: PHP (php)
```

This approach is simple but ties notification logic directly to the controller action. For bigger apps, consider using events.

Trigger Notifications with Events

Fire an event in your controller and handle the notification in a listener:

```
// Listener example
class SendUserNotification
{
    public function handle(UserRegistered $event): void
    {
        $event->user->notify(new UserRegisteredNotification());
    }
}Code language: PHP (php)
```

This decouples notification logic from controllers and lets you attach multiple listeners for additional tasks.

Display Notifications in the Blade UI

Show a user's notifications directly on their dashboard:

```
@foreach (auth()->user()->notifications as $notification)
    <div class="alert alert-info">
        {{ $notification->data['message'] ?? 'New Notification' }}
    </div>
@endforeachCode language: PHP (php)
```

This simple loop displays messages for the authenticated user. You can enhance it with Bootstrap components, icons, and read/unread states.

Testing Notifications with PHPUnit

Fake notifications in tests to verify they are sent correctly without calling external services:

```
Notification::fake();

$user = User::factory()->create();
$user->notify(new UserRegisteredNotification());

Notification::assertSentTo(
    $user,
    UserRegisteredNotification::class
);Code language: PHP (php)
```

`Notification::fake()` prevents delivery, while `assertSentTo()` confirms the notification was dispatched to the intended recipient.

Wrapping Up

Laravel's notification system lets you send messages via email, SMS, and Slack with ease. You learned how to configure providers in `.env`, create a notification class, use them inside controllers, dispatch via events for cleaner architecture, render notifications in the UI, and test them effectively. These skills help you deliver timely communication while keeping code maintainable and scalable.

FAQ: Laravel Notifications

- **How do I send an email notification in Laravel 12?** Use the `notify()` method on a notifiable model and define a `toMail()` method in your notification class.
- **Can I send SMS notifications in Laravel?** Yes. Configure Nexmo or Twilio in `.env` and add `toNexmo()` or `toTwilio()` methods in your notification class.
- **How do I send Slack notifications?** Set `SLACK_WEBHOOK_URL` in `.env` and implement `toSlack()` in your notification class using `SlackMessage`.
- **Should I use controllers or events for notifications?** Controllers are fine for small apps. For scalable apps, dispatch events and let listeners send notifications.

What’s Next

Keep building a robust communication system by checking out these related guides:

- How to Queue Emails in Laravel for Faster Delivery
- Laravel Events and Listeners: A Complete Guide
- Using Laravel Telescope to Debug Performance Issues