

EMAIL · RESEND · SMTP · DNS

Moving your app's email to Resend

Move password resets, notifications and other app emails off your own mail account's SMTP. Domain verification on Resend, DNS records, a locked-down API key, connecting over SMTP or the API, and testing.

TIME	LEVEL	STEPS	PROMPTS
20 min	Intermediate	6	1

The emails your app sends (password resets, sign-up confirmations, notifications) often go out through the SMTP of your own work or personal mail account. That's fine at first. As users grow, so does the risk. This guide walks you through moving those emails to Resend, step by step. I did this for my own personal finance app. Hit **Copy** in the top right of any command or code block, paste it, keep going.

What we're making

We'll separate your app's email from your own mail account and move it to Resend. First, why.

Sending app email through the SMTP of a work account like Google Workspace carries three risks:

1. **Sending limits.** Mail accounts are built for people and have a daily sending cap. As your app grows, that cap can fill up. When it does, both password resets and your own work emails may stop going out.
2. **Reputation.** This is the real risk. If your app and your work email share the same account and domain, every time a user marks an app email as spam, it counts against that domain's reputation. Your normal work emails can then start landing in spam too.
3. **No visibility.** Which email arrived, which bounced, which went to spam? A normal mailbox won't tell you. When a user says "I never got the email", you have nothing to check.

SMTP isn't the problem. SMTP is the set of rules for sending email. Think of it as the traffic rules of a highway: cars and trucks follow the same rules. Resend supports SMTP too. The issue isn't the protocol. It's **separating your app's email from your own mailbox.**

Resend isn't the only option either. Amazon SES and Postmark do the same job. I picked Resend for its simple dashboard and short setup.

You'll end up with three things:

1. Your domain verified on Resend
2. Your app sending its email through Resend (over SMTP or the API)
3. A sending log that shows which emails arrived

INGREDIENTS

- Your own domain and access to its DNS settings (Cloudflare, your registrar, etc.)
- A [Resend](#) account. You can start on the free plan.

- An app that sends email: WordPress, Node.js / Next.js, or anything with SMTP settings.

As of October 2026, the free plan sends 3,000 emails a month and 100 a day. Pro is \$20 a month for 50,000 emails. Numbers change, so check resend.com/pricing for the current limits.

TOOLS

Resend

Cloudflare

WordPress

Node.js

01 Sign up and add your domain

1. Sign up for [Resend](#).
2. Go to **Domains** in the sidebar and click **Add domain**.
3. Type your domain: .
4. Pick a region. If most of your users are in Europe or Turkey, the EU region () makes sense.

The region is chosen when you add the domain. Changing it later may mean deleting and re-adding the domain, so pick the right one up front.

02 Add the DNS records

Once the domain is added, Resend shows a few DNS records (MX, a TXT for SPF, a TXT for DKIM). They're how you say "Resend may send email on

behalf of this domain."

If you use Cloudflare: the dashboard offers an automatic setup for Cloudflare. You sign in to your Cloudflare account, approve, and the records are added in one click.

If you use another DNS provider: enter the records the dashboard shows into your DNS panel one by one. Watch out for two things:

- Don't repeat your domain in the **Name** field. Type just `send`, not `send.yourdomain.com`. Most panels append the domain for you.
- If you're entering them manually in Cloudflare, the records must not be proxied (**DNS only**, grey cloud).

Don't copy record values from this guide. **Enter exactly what you see in your Resend dashboard.** The values differ per account and domain.

After adding the records, click **Verify** in Resend. Verification can take anywhere from a few minutes to a few hours. Mine took about 20 minutes.

Extra tip: a DMARC record. DMARC tells receiving servers what to do when SPF or DKIM fails. If you don't have one, add it:

TEXT

Type: TXT
Name: `_dmarc`
Value: `v=DMARC1; p=none;`

If your domain already has a `_dmarc` record, don't add a second one. A domain has only one DMARC record.

03 Create an API key

1. Go to **API Keys** in the sidebar and click **Create API key**.
2. Give it a name (e.g. `app-prod`).
3. **Permission**: choose **Sending access**, not Full access. Your app only needs to send email; if the key leaks, nobody should be able to manage your account with it.
4. **Domain**: restrict it to this domain only.

The domain may not show up in the list until it's verified. In that case, create the key for now, then once verification is done, create a new key restricted to the domain and delete the old one.

The key starts with `re_` and is **shown only once**. Save it in your password manager right away. Don't put it in code, in your repo or in a screenshot. On the server, keep it in an environment variable:

TERMINAL

```
RESEND_API_KEY=re_xxxxxxxx
```

04 Connect your app

Two ways. Use SMTP if you barely want to touch your code, the API if you're on Node.js / Next.js.

Over SMTP (almost no code changes)

Change your app's SMTP settings to:

TEXT

```
Host:      smtp.resend.com
Username:  resend
Password:  your API key (re_...)
Port:      465 (SSL) or 587 (STARTTLS)
```

The sender address must be on your verified domain, e.g.

`notifications@yourdomain.com`. You can't send from another domain.

WordPress example. First, read the key from an environment variable in `wp-config.php`:

PHP

```
define( 'RESEND_API_KEY', getenv( 'RESEND_API_KEY' ) );
```

Then add this to your theme's `functions.php` or a small plugin:

PHP

```
add_action( 'phpmailer_init', function ( $phpmailer ) {  
    $phpmailer->isSMTP();  
    $phpmailer->Host      = 'smtp.resend.com';  
    $phpmailer->SMTPAuth  = true;  
    $phpmailer->Username  = 'resend';  
    $phpmailer->Password  = RESEND_API_KEY;  
    $phpmailer->Port      = 465;  
    $phpmailer->SMTPSecure = 'ssl';  
    $phpmailer->From      = 'notifications@yourdomain.com';  
    $phpmailer->FromName  = 'Your App';  
} );
```

Every email WordPress sends now goes out through Resend.

Over the API (Node.js / Next.js)

Install the package:

TERMINAL

```
npm install resend
```

Send an email:

JAVASCRIPT

```
import { Resend } from 'resend';

const resend = new Resend(process.env.RESEND_API_KEY);

const { data, error } = await resend.emails.send({
  from: 'Your App <notifications@yourdomain.com>',
  to: ['user@example.com'],
  subject: 'Reset your password',
  html: '<p>Click the link to reset your password.</p>',
});

if (error) console.error(error);
```

You can also have Claude Code do this for you:

PROMPT

I'm moving all the email this project sends to Resend.

1. Find every place in the project that sends email (SMTP settings, mail library, wp_mail, nodemailer, etc.) and list them for me.
2. If it uses SMTP, change the settings to smtp.resend.com, username "resend", port 465. Read the password from the RESEND_API_KEY environment variable.
3. Set the sender address to notifications@<MY DOMAIN>.
4. Never write the API key to any file or commit it. Check that the .env file is in .gitignore.

Show me the plan before making any changes.

05 Test it

1. Request a password reset email from your app to yourself.
2. Check the **Emails** (sending logs) section in Resend. The email should show **Delivered**.
3. Open the email in Gmail. From the **:** menu in the top right, choose **Show original**.
4. **SPF**, **DKIM** and **DMARC** should all say **PASS**.
5. The sender should be on your new domain (`notifications@yourdomain.com`), not your old mail account.

06 Afterwards

- If everything runs smoothly for a few days, revoke the old SMTP credentials (app password, etc.). An unused password is an open door.
- As volume grows, keep an eye on your plan's daily and monthly limits.
- Hook up webhooks for **bounced** and **complained** (marked as spam) events. Repeatedly emailing an invalid address hurts your domain's reputation; remove those addresses from your list.

TIPS

"Domain is not verified" error. You're sending before verification is done. Check the domain's status in Resend and wait for **Verified**.

Verification never finishes. You may have typed the domain twice in the Name field (`send.yourdomain.com.yourdomain.com`). Check the record's

full name in your DNS panel and fix it to just `send` if needed.

The record won't add or verify on Cloudflare. The record may be proxied. Turn off the orange cloud and set it to **DNS only**.

Emails arrive but land in spam. Look at DKIM and DMARC in "Show original". If they don't PASS, compare your DNS records with the ones in the Resend dashboard. Add a DMARC record if you don't have one.

DONE?

- ☐ Added my domain in Resend and picked a region
- ☐ Added the DNS records, the domain is **Verified**
- ☐ (Recommended) There's a `_dmarc` record
- ☐ Created an API key with **Sending access** only, restricted to the domain
- ☐ The key is in my password manager and in an env variable on the server, not in the repo
- ☐ Connected the app over SMTP or the API
- ☐ Test email shows SPF, DKIM, DMARC: PASS
- ☐ Revoked the old SMTP credentials after a few days

Got stuck somewhere? Tell me on Instagram and I'll add it to the guide.