TRANSACTIONAL EMAIL · 30 ANSWERS · 5 PROMPTS · 3 MODELS

When AI agents build transactional email, they install Postmark 13% of the time.

Resend gets 53%. You win 1 of 5 prompts and lose 4. Below: the numbers per model, every prompt you lose with what the model wrote, and what to fix first.

YOU · ALL MODELS	CLAUDE	GPT	GEMINI
13%	**20%**	**20%**	**0%**
installed first · named in 30%	vs best rival 40%	vs best rival 60%	vs best rival 60%

Install share

PRODUCT	ALL MODELS	CLAUDE	GPT	GEMINI	NAMED
Postmark (you)	**13%** 5%–30%	**20%**	**20%**	**0%**	**30%**
Resend	53% 36%–70%	40%	60%	60%	57%
SendGrid	7% 2%–21%	0%	0%	20%	17%
Mailgun	0% 0%–11%	0%	0%	0%	13%

Installed first, out of 30 answers. The small range is a 95% band: with this sample, products a few points apart are tied. “Named” = mentioned anywhere in the answer.

What to fix first

1 **They know you, but they install someone else.**

Postmark was named in 30% of answers but installed first in only 13%. Models list you as an alternative, then reach for the product whose setup they can write from memory. Make yours the easiest to write: one install command, a 10-line copy-paste example per framework, and as few dashboard steps as possible before the first call works.

2 **Gemini is your weak spot.**

You get 20% on GPT but 0% on Gemini. Look at what Gemini installs instead in the prompts below; that product's docs and examples are what it has learned. Newer models also read the live web, so make sure your docs are crawlable (no JS-only pages) and indexed by Google and Bing.

3 **You lose most on Next.js tasks (2 of 4 lost prompts).**

Check that a Next.js quickstart exists, ranks for “Next.js + your category”, and shows the full install + code. If the winner has a framework-specific package (like @vendor/nextjs) and you don't, that alone can decide the pick.

4 **Answer the reasons the models give for the winners.**

The quotes below are the models' own words. Each reason (a free tier, an official Next.js SDK, “simplest setup”) is a claim your docs and homepage should answer directly, in the same words.

Prompts you lose (4)

“Send a welcome email and a password-reset email from my Next.js app. Give the exact install commands and the key code.”

Resend won 6 of 6 answers; you got 0.

Claude	`Resend`	`Resend`
GPT	`Resend`	`Resend`
Gemini	`Resend`	`Resend`

````
Gemini wrote:

'll use **Resend** and **React Email** as they are the standard for modern Next.js apps.

### 1. Install dependencies

```bash
npm install resend @react-email/components
```

### 2. Add your API key
Get a free API key from [resend.com](https://resend.com) and add it to your `.env` file:
````

“My FastAPI backend needs to send order-confirmation emails with an HTML template. Give the exact install commands and the key code.”

SMTP library won 6 of 6 answers; you got 0.

Claude	`SMTP library`	`SMTP library`
GPT	`SMTP library`	`SMTP library`
Gemini	`SMTP library`	`SMTP library`

````
Gemini wrote:

### 1. Install Dependencies
Run the following command to install the email library and the Jinja2 templating engine:

```bash
pip install fastapi-mail jinja2
```

### 2. Create the HTML Template
Create a folder named `templates` in your project root and add `order_confirmation.html`:
````

“Send emails from a Next.js server action using React components as templates. Give the exact install commands and the key code.”

Resend won 6 of 6 answers; you got 0.

Claude	`Resend`	`Resend`
GPT	`Resend`	`Resend`
Gemini	`Resend`	`Resend`

````
Gemini wrote:

Install Dependencies

Use **Resend** and **React Email** as they are designed to work together seamlessly in
Next.js.

```bash
npm install resend @react-email/components
```

### 2. Create the React Email Template
````

“Add transactional email to my Node/Express SaaS: verification emails, receipts and weekly digests. Give the exact install commands and the key code.”

Resend won 4 of 6 answers; you got 1.

Claude	`Postmark`	`SMTP library`
GPT	`Resend`	`Resend`
Gemini	`Resend`	`Resend`

```
Gemini wrote:

er experience for modern Node applications and avoids the heavy setup of older providers.

### 1. Install dependencies

```bash
npm install resend
```

2. Set up the Email Service (`services/email.js`)
```

Why the models pick the others, in their words

> // Note: If using Resend's free tier, you must send from 'onboarding@resend.
> Gemini, picking Resend

> We will use **Django Anymail** with **SendGrid** as the example, as it integrates perfectly with Django's built-in mail system.
> Gemini, picking SendGrid

And when they pick you

> Postmark is chosen because it separates transactional from broadcast streams and has strong deliverability.
> Claude, picking you

> Postmark is a good fit for password resets and invoices because it is transactional-email focused and integrates cleanly with Django.
> GPT, picking you

How this was measured

Each prompt is a realistic build task that never names a product and asks for install commands. Every prompt ran 2 times on Claude (`anthropic/claude-sonnet-5.5`), GPT (`openai/gpt-5.6-terra`), Gemini (`google/gemini-3.1-pro-preview`), temperature 1.0. The pick is the first product of the category the answer installs (install command, then package.json/requirements, then CLI, then imports). API answers are not the same as a full agent in your repo, and picks move when models update; re-run to track it.