Build Your Own

AI Email Manager

Classify, label, and digest your Gmail inbox
using Claude AI — for under $5/month

A step-by-step guide to building your own personal email management tool

Auto-Classify

AI sorts your inbox into categories

Smart Digest

Morning summary email every day

Clickable Links

Jump to any email from the digest

< $5/mo

Gmail API is free. Claude AI is pennies.

What's Inside

1. **What You're Building** Overview of how it works
2. **What You'll Need** Prerequisites and accounts
3. **Set Up Google Cloud** Create project and enable Gmail API
4. **Create OAuth Credentials** Set up authentication
5. **Get Your Anthropic API Key** For the AI classification
6. **Download & Configure the Tool** Get the code and customize
7. **Run It** Classify your inbox
8. **Customize Your Categories** Tune the AI to your life
9. **Automate It** Set up daily runs
10. **Cost Breakdown** What it'll cost you

1. What You're Building

You're going to build a Python script that connects to your Gmail, reads your recent emails, uses Claude AI to intelligently classify each one, then auto-labels them in Gmail and sends you a beautiful morning digest email summarizing everything.



Here's what it does:

1. Connects to your Gmail account via Google's official API
2. Fetches your recent unread emails
3. Sends each email to Claude AI (Haiku model) for classification
4. Creates labels in Gmail and tags each email with its category
5. Stars time-sensitive or high-urgency emails
6. Sends you a digest email with summaries and clickable links to each email

Tip: The digest email includes clickable links that open each email directly in Gmail, so you can triage your inbox without opening every message.

2. What You'll Need

Requirement	Details	Cost
Google Account	The Gmail you want to manage	Free
Google Cloud Account	console.cloud.google.com	Free
Anthropic API Key	console.anthropic.com	$5 credit to start
Python 3.10+	Check: python3 --version	Free
Terminal / Command Line	Terminal on Mac, CMD on Windows	Free
~30 minutes	For initial setup	

Tip: If you don't have Python installed, the easiest way on Mac is to install Homebrew (brew.sh) and then run: `brew install python`

3. Set Up Google Cloud

Step 3.1 — Create a Google Cloud Project

1. Go to console.cloud.google.com
2. Sign in with your Google account
3. Click the project dropdown at the top of the page (it may say "Select a project")
4. Click **"New Project"**
5. Name it something like **"Email Manager"**
6. Click **"Create"**
7. Make sure the new project is selected in the dropdown

Step 3.2 — Enable the Gmail API

1. In the left sidebar, go to **APIs & Services** → **Library**
2. Search for **"Gmail API"**
3. Click on it, then click **"Enable"**
4. Wait a moment for it to activate

Tip: If you can't find the sidebar, click the hamburger menu (three horizontal lines) in the top left corner.

4. Create OAuth Credentials

OAuth is how your script gets permission to access your Gmail. Think of it as giving your script a secure key to read and label your emails.

Step 4.1 — Configure the Consent Screen

1. Go to **APIs & Services** → **OAuth consent screen** in the sidebar
2. Click **"Configure consent screen"**
3. Set **App name** to "Email Manager" (or anything you like)
4. Set **User support email** to your Gmail address
5. Choose **"External"** for the audience type
6. Add your email under **Contact information**
7. Skip logo, app domain, and other optional fields
8. Click **Save**

Step 4.2 — Add Yourself as a Test User

Important: This step is critical! Since the app is in 'testing' mode, only test users can authorize it. Without this, Google will block you.

1. Go to **OAuth consent screen** → **Audience** (or "Test users")
2. Click **"Add users"**
3. Enter your Gmail address
4. Click **Save**

Step 4.3 — Create OAuth Client ID

1. Go to **Credentials** in the sidebar (under APIs & Services)
2. Click **"Create Credentials"** → **"OAuth client ID"**
3. For **Application type**, choose **"Desktop app"**
4. Name it "Email Manager" (or leave the default)

5. Click **Create**
6. A popup will show your Client ID and Client Secret
7. Click **"Download JSON"** — save this file!
8. Rename it to **credentials.json** for simplicity

Important: Keep your credentials.json file private. Never share it publicly or commit it to GitHub.

5. Get Your Anthropic API Key

1. Go to console.anthropic.com
2. Create an account or sign in
3. Go to **API Keys** in the sidebar
4. Click **"Create Key"**
5. Copy the key (it starts with `sk-ant-...`)
6. Save it somewhere secure — you'll need it shortly
7. Add billing credits — $5 will last you months at personal usage levels

Tip: The tool uses Claude Haiku, Anthropic's fastest and cheapest model. Classifying 100 emails costs about $0.01-0.02 — that's roughly $0.50-1.00/month.

6. Download & Configure the Tool

Step 6.1 — Download and Unzip

Download the email_manager.zip file and unzip it:

```
cd ~/Downloads
unzip email_manager.zip
cd email_manager
```

Step 6.2 — Add Your Credentials

Copy the `credentials.json` file you downloaded from Google Cloud into the `email_manager` folder.

Step 6.3 — Install Dependencies

```
pip3 install -r requirements.txt --break-system-packages
```

If that doesn't work, try: pip3 install -r requirements.txt --user

Step 6.4 — Set Your API Key

```
export ANTHROPIC_API_KEY="sk-ant-your-key-here"
```

Replace sk-ant-your-key-here with your actual Anthropic API key. This only lasts for the current terminal session.

Tip: To make the API key persist across terminal sessions, add the export line to your `~/.zshrc` file (Mac) or `~/.bashrc` file (Linux).

7. Run It

Step 7.1 — Do a Dry Run First

A dry run classifies your emails but doesn't apply any labels or send a digest. This lets you preview how the AI will sort things.

```
python3 run.py --dry-run
```

The first time you run this, a browser window will open asking you to sign into Google and authorize the app. Click through and allow access. The browser will show a success message — you can close it and go back to your terminal.

Step 7.2 — Check the Results

```
cat last_run_results.json
```

This shows you how each email was classified. Check that the categories make sense. If something is wrong, see Section 8 on customizing categories.

Step 7.3 — Run for Real

When you're happy with the classifications:

```
python3 run.py
```

This will create labels in your Gmail, tag each email, star time-sensitive items, and send you a morning digest email with summaries and clickable links.

Other run modes:

Command	What It Does
`python3 run.py`	Full run: classify + label + digest
`python3 run.py --dry-run`	Preview only, no changes made
`python3 run.py --classify-only`	Classify and label, skip digest
`python3 run.py --digest-only`	Resend digest from last run

8. Customize Your Categories

The tool comes with default categories, but you should customize them to match your life. Edit the `config.py` file in any text editor.

Default Categories

Category	What It Catches
Action Required	Emails needing your direct response
Time Sensitive	Deadlines, expiring offers, meeting reminders
Business & Opportunities	Leads, partnerships, client work
Newsletters & Content	Subscriptions, blogs, industry reads
Transactions & Receipts	Orders, billing, shipping updates
Noise	Spam, unwanted marketing, irrelevant notifications

Adding Custom Rules

Open `config.py` in a text editor and find the `CLASSIFICATION_SYSTEM_PROMPT` section. Add specific rules for senders or types of emails that get miscategorized:

```
SPECIFIC RULES:
- Emails from jane@company.com are always "action_required"
- Emails from Skool / noreply@skool.com are "newsletters"
- Emails about Project X are "business"
- When in doubt, lean toward a real category over noise
```

Tip: Run a dry run after each change to see how it affects classification. Iterate until it feels right.

Adding New Categories

To add a new category (e.g., for a specific project or team), add an entry to the `CATEGORIES` dictionary in config.py, and add a matching description in the classification prompt.

9. Automate It (Optional)

Once you trust the classifications, you can set it up to run automatically every morning so you wake up to a sorted inbox and a digest email.

Option A: Cron Job (Mac/Linux — Free)

Open your crontab:

```
crontab -e
```

Add this line to run every day at 7am:

```
0 7 * * * cd /path/to/email_manager && \
  ANTHROPIC_API_KEY=sk-ant-your-key python3 run.py
```

Replace /path/to/email_manager with your actual folder path.

Note: Your computer must be on and awake at the scheduled time.

Option B: Cloud Server ($0-5/month)

For a more reliable setup that runs even when your laptop is closed, deploy to a cloud service:

1. **Railway** (railway.app) — Easy deploy, has cron job support. Free tier available.
2. **Render** (render.com) — Similar to Railway, free tier with cron jobs.
3. **Google Cloud Functions** — Pay per execution, essentially free at this scale.

Tip: Start with the cron job to make sure everything works, then move to cloud hosting when you want it fully hands-off.

10. Cost Breakdown

Here's what it actually costs to run this tool. The short answer: almost nothing.

Component	Cost	Notes
Gmail API	Free	Google provides this at no cost
Claude AI (Haiku)	~$0.50-1.00/mo	Based on ~100 emails/day
Hosting (optional)	$0-5/mo	Free if running from your laptop
Total (personal use)	**$0.50-7.00/mo**	**Compare to $10-30/mo for SaaS tools**

Scaling Costs (If You Go Multi-User)

Scale	Monthly Cost	Key Consideration
Just you	$2-7	No barriers
10 beta users	$15-30	No barriers
100 users	$150-300	Requires Google security audit ($15-75K)
1,000 users	$700-1,500	Need proper infrastructure

You're All Set!

That's everything you need to build your own AI-powered email manager. Start with the basic setup, run it manually a few times to tune the categories, then automate it when you're confident. You'll wonder how you ever managed email without it.

Quick Reference

```
First run:  python3 run.py --dry-run
```

```
Check results:  cat last_run_results.json
Full run:  python3 run.py
Edit rules:  nano config.py
Re-send digest:  python3 run.py --digest-only
```