

Bulk Email Outreach via Gmail API

Sanitized step-by-step setup and usage guide (no personal data).

What this project does

A local Python CLI tool that sends templated outreach emails with an optional PDF attachment using the Gmail API and OAuth 2.0. It supports rate limiting and a dry-run mode to validate inputs before sending.

Security rules (must follow)

Never commit secrets or personal data to GitHub. Keep OAuth credentials, token files, email lists, and resumes on your local machine only. Use a `.gitignore` file to prevent accidental commits.

Local-only files (do not commit)

- `credentials.json` (OAuth client JSON downloaded from Google Cloud)
- `token.json` (created automatically after first OAuth approval)
- `emails.txt` (recipient list — one email address per line)
- `Resume.pdf` (or any attachment PDF you choose)

Repository contents (safe to commit)

- `gmail_bulk_send_oauth.py` (main CLI script)
- `requirements.txt` (Python dependencies)
- `README.md` and `docs/` (project documentation)
- `.gitignore` (prevents secrets and personal files from being committed)

Step-by-step instructions

1) Prerequisites

- Python 3.8+ installed (3.12 OK).
- A Gmail account to send from (the sender mailbox).
- A Google Cloud project (any name).

2) Local setup

- Clone the repository to your machine.
- Create and activate a virtual environment.
- Install dependencies from requirements.txt.

3) Create local input files

- Create emails.txt with one email per line.
- Place your attachment PDF (e.g., Resume.pdf) in the project folder.

4) Google Cloud setup (one-time)

- Enable Gmail API in the selected Google Cloud project.
- Configure OAuth consent screen as External and set publishing status to Testing.
- Add your sender Gmail address as a Test User.
- Create OAuth Client ID (Desktop app) and download credentials.json into the project folder.

5) Run the tool

- Run a dry-run first to preview subject/body and validate files.
- Send a small test batch (e.g., 3-5 emails).
- Increase max emails only after the test is successful.

6) First-time OAuth approval

- On first real send, a browser window opens.
- Sign in with the Gmail account you want to send from.
- Approve access; token.json will be created locally.

7) Operational safety

- Use a delay (2-5 seconds) between sends.
- Send in batches (5 → 20 → 50) rather than blasting hundreds.

- Keep messages simple to reduce spam flags.

8) Troubleshooting

- Missing credentials.json: download OAuth Desktop credentials and place in the project folder.
- Access blocked in Testing: ensure the sender Gmail is added as a Test User on the OAuth consent screen.
- Wrong sender authorized: delete token.json and run again to re-authorize.

Command reference (examples)

Install dependencies

```
python3 -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
```

Dry-run (no emails sent)

```
python3 gmail_bulk_send_oauth.py --emails emails.txt --resume "Resume.pdf" --dry-run
```

Send a small test batch

```
python3 gmail_bulk_send_oauth.py --emails emails.txt --resume "Resume.pdf" --max-emails 5 --delay 3
```

Send to all recipients

```
python3 gmail_bulk_send_oauth.py --emails emails.txt --resume "Resume.pdf" --max-emails 0 --delay 3
```

Re-authorize sender account

```
rm -f token.json
python3 gmail_bulk_send_oauth.py --emails emails.txt --resume "Resume.pdf" --max-emails 3
```

Notes

This document intentionally omits personal identifiers such as email addresses, resumes, and tokens. Store those locally and ensure they are excluded via .gitignore.