AUTOMATIZE · COMMUNICATION DELIVERY

Out boundly.

A transactional email API in front of your own mail server — and everything that makes it actually deliver.

Your applications POST JSON and get an audited send. Behind that sits the operational work nobody wants to do by hand: domain ownership verification, DKIM key generation, ready-to-publish DNS records, mail-server map generation and reload, TLS certificates that renew themselves, and a queue and log console for the day something is stuck.

Self-hosted | **Postfix operations console** | **DKIM · SPF · DMARC** | **Automatic TLS** | **Made in Europe**

90	0	11	1730
DAY RETENTION CEILING	MESSAGE BODIES STORED	CLIENT CODE LANGUAGES	AUTOMATED TESTS

SOFTWARE VENDORS · MSPS · HOSTING PROVIDERS · PRIVACY-LED BUYERS

PRODUCT BROCHURE 2026.1

THE PROBLEM

Two bad options: hand your mail to a foreign service, or run Postfix by hand.

The hosted senders are easy and put every password reset, invoice and notification your customers receive through infrastructure in someone else's jurisdiction, priced per thousand. Running your own Postfix keeps the data but hands your team DKIM keys, DNS records, virtual maps, certificate renewals and a queue that occasionally backs up at three in the morning.

Data leaves
Every recipient address and subject line processed outside your control.

Priced per thousand
A bill that grows with the thing your product does automatically.

Deliverability by hand
DKIM, SPF and DMARC set up once by someone who has since left.

No answer at 3am
Mail is stuck and the only tool is an SSH session and a man page.

THE SOLUTION

Keep your own mail server. Put a proper API and an operator console on top of it.

The credential is the boundary
An API key is scoped to specific mailboxes. The sending address is **never a request parameter** — so there is nothing for a compromised caller to lie about.

Audited before it is sent
The audit row is written **before** the transport is asked, in its own transaction. A failed send is a logged failure, never an absent row.

Bodies are never stored
Addresses, subject, counts and outcome — never message content. And retention is a **ceiling in the code**, not a setting somebody can quietly raise.

GETTING TO THE FIRST SEND

1. **Register the application**
   A guided five-step wizard walks from application, to domain, to DNS verification, to mailbox, to a test send that proves the whole path.
2. **Prove the domain**
   Publish a generated verification record; ownership is checked over both TXT and MX independently. Then generate a DKIM key and publish the SPF, DKIM, DMARC and MX records the product hands you.
3. **Create the mailbox**
   The product generates the mail-server maps, validates the configuration and reloads it. Per-mailbox reply-to, copy and blind-copy policy is set here, not by the caller.
4. **Send**
   POST JSON with an API key. Delivery is retried with exponential backoff; a refused relay comes back as a 502 carrying the audit log identifier, so the failure is traceable rather than opaque.

SEND & DELIVER

The API, and the deliverability work behind it.

1 The email API

- A JSON send endpoint served under **two version prefixes**, with a build-failing test that stops a controller silently dropping off the older one
- Recipients, copies, subject, HTML body, attachments and reply-to, with per-mailbox defaults applied on every send
- Limits enforced in three independent layers: **20 recipients, 20 copies, 5 attachments, 10 MB each, 25 MB total** — over the limit is a rejection, never a silent truncation
- Attachment types recognised by extension; filenames rejected if they carry path or shell characters
- **No blind-copy request field, by design** — a blind copy is a mailbox policy, not something a caller sets
- Delivery retried three times with exponential backoff; permanent failures are not retried
- Header injection defended by stripping line breaks from every header-bound value

2 Domains, DNS & DKIM

- Ownership verified over **TXT and MX independently**, each tracked separately, with an administrator override that is enforced in the service rather than the screen
- Verification tokens generated securely; lookups performed directly against DNS
- **DKIM key generation** at 2048 bits, with the signing tables maintained for you — and the private key never entering the application or the database
- Ready-to-publish **SPF, DKIM, DMARC and MX records**, presented in tabbed modals with per-field copy buttons
- A guided verification page including a panel of common DNS provider instructions

3 Credentials & access

- API keys generated securely, **stored only as a hash**, shown in plaintext exactly once
- Revoked rather than deleted, so historical sends keep their audit trail intact
- **Scoped to specific mailboxes** through a join table, managed in a two-pane assignment screen
- Per-user, per-mailbox grants for primary, send and manage rights
- Mailbox passwords hashed in the format the mail server itself expects
- Rate limiting on authentication, registration, validation and test paths, returning a retry hint — with the newer version prefix sharing one bucket with the older, so a limit belongs to the operation rather than the URL

4 TLS, automatically

- Certificates issued and renewed over the standard automated protocol
- **Both challenge types**: an HTTP challenge served by the application, or a DNS challenge published through a supported provider or manually, with the record to publish shown and the flow resumable
- Renewal on a schedule, by default thirty days before expiry
- Issued material installed into the mail server at the configured paths
- Every certificate records issue, expiry and renewal dates, serial, issuer, challenge type, last error and attempt count

> **Why the sending mailbox is not a parameter.** Most email APIs let the caller declare who the message is from and check it against a list. Outboundly resolves the mailbox from the credential itself. A leaked key can send only from the mailboxes it was granted — which is a materially smaller blast radius than a leaked key that can send as anyone on the domain.

OPERATE & ACCOUNT FOR IT

The operator console and the audit record.

5 Mail-server operations

- **Map generation and reload** — domains and mailboxes rendered from the database, written, compiled, validated and reloaded, on demand or on a schedule
- **Queue management**: list and filter the queue; hold, release, requeue or delete a message; flush, requeue, hold, release or delete in bulk; delete by sender or recipient pattern
- Queue statistics, per-directory counts, and summaries by sender and by recipient
- Full message inspection — envelope, headers and body — for anything stuck
- **Configuration file editor** confined to a directory allow-list, with search, built-in templates, timestamped backups, validate, reload and restart
- Command-injection defences throughout: allow-listed commands validated at start-up, and pattern-constrained queue identifiers, filenames and log paths

6 Log console

- Search the mail, error and warning logs and the system log
- **Rotated and compressed files searched too** — the answer is usually in yesterday's log, not today's
- Wildcard patterns, time-span filtering and multi-format timestamp parsing
- Live tail; download the search results or the whole log

7 The audit record

- A row for **every send attempt**, written before the transport is asked, in its own transaction — so the row survives the exception that caused the failure
- Records the key, application, mailbox, addresses, subject, counts, time, status, attempt count and error — and **never the message body**
- Attachments archived by date and log identifier, for failures as well as successes, without ever failing a delivered send
- **Retention clamped to ninety days in code**, logged when a larger value is requested — a promise the configuration file cannot quietly break
- Purge runs in bounded batches, deleting rows before files and then sweeping whole date directories regardless of database state
- Query by key, application, mailbox, status and date range; per-organisation daily, weekly and monthly totals on the dashboard
- Created and modified stamps plus optimistic locking on every record

8 Developer experience

- An **in-product API explorer** — operations grouped and filterable, parameter and schema tables, and a live try-it form riding your own session
- **Generated client code in eleven languages**, including COBOL, Fortran and RPG, with twenty-five more samples on the public site
- Every specification string reaches the page as text, so a description containing markup stays a description
- An SMS queue with destination-prefix rules and an API for an external gateway to collect and report on messages
- A memory console with live heap, a threshold-driven schedule and a manual trigger

Where the product is today. Outboundly is a mail platform, and that is what we sell. Its subscription and billing subsystem is not production-ready, quota enforcement is not wired to the plans shown in the interface, and SMS is a queue with a gateway callback API rather than a sending integration. Deploy it behind your own access layer, and talk to us before you position it on billing.

WHO BUYS IT

Anyone who sends mail from software and cannot outsource it.

Software vendors
Password resets, notifications and receipts sent from your product — with per-application, per-mailbox credentials so one customer's key cannot send as another.

Managed service providers
Transactional mail on behalf of many client applications, from infrastructure you already run and already bill for.

Hosting providers
The queue, log and configuration console is an operator tool. It turns mail support from an SSH session into a screen a junior can use.

Privacy-led organisations
No message body is ever stored and the audit window is a hard ninety days — a defensible answer rather than a policy document.

Public sector
Self-hosted and European, where sending citizen correspondence through a foreign service is not procurable.

Healthcare-adjacent
Appointment and result notifications where the recipient list itself is sensitive and must stay inside the estate.

Financial services
Statement and alert delivery with a per-send audit record and a signed, verifiable sending domain.

Belgian & Benelux business
Built here: European hosting, European defaults, and an operator model that assumes you own the mail server.

Legacy integration shops
Client code generated for COBOL, Fortran and RPG alongside the modern languages — a deliberate signal to mainframe and midrange estates.

WHO USES IT

Platform administrator · **Organisation user** · **Application / machine caller** · **External SMS gateway**

The machine caller has no account at all — it authenticates with a key and touches exactly one endpoint. Per-mailbox grants let an organisation delegate send and manage rights without inventing another role.

AGAINST THE ALTERNATIVES

OPTION	WHY IT FAILS HERE
A hosted sending service	Priced per thousand; recipients and subjects processed off-site
Raw SMTP from the application	No audit, no key scoping, no retry policy, no operator view
Postfix administered by hand	DKIM, DNS, maps and certificates are all somebody's memory
A shared relay account	One credential that can send as anyone, and no way to revoke narrowly

TECHNICAL SPECIFICATION

A JAR, a service unit, and the Postfix you already have.

Runtime	Java 21 on Spring Boot
Database	PostgreSQL
Mail transport	Your own Postfix — the product configures and operates it rather than replacing it
Front end	Server-rendered with hand-written JavaScript — no package manager, no bundler, and **no inline script in the admin shell**
API	26 controllers served under two version prefixes, with an OpenAPI document and an in-product explorer
Certificates	Standard automated issuance with HTTP and DNS challenge support and scheduled renewal
Deployment	Single node, service unit and reverse proxy, documented in a full installation guide including a clustered mail topology
Scale	~105 000 lines first-party across application, templates, script and styles
Quality	**1 730 tests** across 111 classes — roughly one line of test per line of production code

SECURITY & GOVERNANCE

- API keys hashed at rest, revealed once, revocable, and **scoped to individual mailboxes**
- bcrypt for user passwords; the mail server's own hash format for mailboxes
- **Rate limiting** on login, registration, validation and test paths, with the retry hint returned to the caller
- Request-forgery protection on the administration pages, with the fetch and interaction layers patched to carry the token
- Cookies hardened; header injection blocked on every header-bound parameter
- **Command-injection defences**: allow-listed shell commands validated at start-up, pattern-constrained identifiers and a directory allow-list for configuration edits
- Mass-assignment defences on the wizard forms and on registration
- Audit rows written ahead of the send, and a **ninety-day retention ceiling clamped in code**
- Dependency vulnerability scanning available in the build, with several fixes pinned explicitly

Not in this release: multi-factor authentication, single sign-on, encryption at rest, and data-subject export or erasure tooling. The administrator role is global rather than per-tenant, and organisation separation is by column rather than enforced in the database. Deploy behind your own access layer on a trusted network.

COMMERCIAL

Licensed per installation. Not per thousand.

The whole argument against a hosted sender: volume stops being a cost line. You run the mail infrastructure, you own the sending reputation, and the send button costs the same in a quiet month and a campaign month.

REVENUE	PER YEAR
Up to €500k	€900
€1M	€1,500
€5M	€2,500
€25M	€5,500

REVENUE	PER YEAR
€100M	€13,000
€500M	€37,000
€1B	€52,000

Annual licence on your own turnover, ex VAT. The schedule is marginal like income tax — each slice of turnover is charged at its own rate, so there are no cliffs — with a €900 minimum and any yearly increase capped at 25%. Every tranche carries unlimited users, projects and organisations inside one tenant, self-hosting and support that rises with the tranche. Hosting and hardware are yours, and consultancy, setup, migration and training are quoted separately. Illustrative: a hosted sender at €0.80 per thousand messages bills a twenty-million-message year €16,000; at €5M turnover this is €2,500.

NEXT STEP

Point one domain at it.

Verify it, generate the DKIM key, publish the four records we hand you, and send one message. It takes an afternoon.

Sales	tim@automatize.eu
Partners	luc@automatize.eu
Web	automatize.eu

Postfix is independent software that Outboundly configures and operates; it is not included or endorsed. Specifications may change.