```
module: 9
title: Email Inbox + Sync (Gmail/Outlook)
phase: 1
status: detail-spec-draft
last_updated: 2026-05-08
session_source: brainstorming-session-2026-05-07-2206.md
related_lps: [LP-7, LP-9, LP-10, LP-11, LP-13]
spec_style: requirements-pur
```

Modul 9 — Email Inbox + Sync

> ***Detail-Spec gemäß LP-9.*** *Verbindet die Mailbox(en) eines Tenants (Gmail / Outlook) mit dem CRM. Jede eingehende Mail wird mit Kontakten/Deals/Projekten verknüpft und im Activity-Stream sichtbar. Liefert das Sending-Domain-Fundament (DKIM/SPF/DMARC) für Modul 10. Bildet zusammen mit M10 die Ablöse-Komponente für HubSpot-Mail-Tools (LP-7).*

Themenblöcke

```
A. Datenmodell
B. Mailbox-Connect (OAuth Gmail / Microsoft)
C. Sending-Domain (DKIM/SPF/DMARC) als Plattform-Fundament
D. Sync-Strategie (Push-Webhooks statt Polling)
E. Auto-Match-Logik (Mail → Kontakt / Deal / Projekt)
F. Inbox-View (Liste + Detail)
G. Outbound-Compose
H. Activity-Stream-Integration
I. Permissions
J. Provider-Adapter (LP-11)
K. Phase 2
```

Block A — Datenmodell

EmailIntegrationMailbox

Pro Tenant-User eine verbundene Mailbox. Tenant-scoped (RLS).

- `id`, `tenant_id`, `created_at`, `updated_at`
- `owner_user_id` (FK auf TenantUser — wem gehört die Mailbox)
- `provider` (Enum: `gmail | microsoft`)
- `email_address` (unique pro Tenant)
- `display_name`
- `oauth_access_token_encrypted`, `oauth_refresh_token_encrypted`, `oauth_token_expires_at`
- `oauth_scope_granted` (Liste der gewährten Scopes)
- `sync_status` (Enum: `connecting | active | paused | error | revoked`)
- `last_sync_at`, `last_error_text`
- `webhook_subscription_id` (Provider-spezifisch — Gmail: Pub/Sub-Topic, Microsoft: Subscription-ID)
- `webhook_expires_at` (Microsoft-Subscriptions laufen nach 3 Tagen ab → Renewal-Job)
- `is_default_for_user` (Boolean — falls User mehrere Mailboxen hat)

- `archived_at`

Indizes: `(tenant_id, owner_user_id)`, `(tenant_id, email_address)`, `(sync_status, webhook_expires_at)` für Renewal-Worker.

EmailIntegrationMessage

Spiegel der Mails aus der angebundenen Mailbox. Tenant-scoped.

- `id`, `tenant_id`, `mailbox_id`, `created_at`
- `provider_message_id` (Gmail-`id` / Microsoft-`internetMessageId`) — unique pro Mailbox
- `provider_thread_id` (Gmail-`threadId` / Microsoft-`conversationId`)
- `direction` (Enum: `inbound | outbound`)
- `from_address`, `from_name`
- `to_addresses[]`, `cc_addresses[]`, `bcc_addresses[]`
- `subject`
- `body_html`, `body_text`
- `received_at`, `sent_at`
- `is_read` (gespiegelt aus Provider)
- `is_starred` (gespiegelt)
- `provider_labels[]` / `provider_folders[]` (z.B. `INBOX`, `SENT`, `Drafts`)
- `has_attachments` (Boolean)
- `matched_contact_id` (FK, nullable)
- `matched_company_id` (FK, nullable)
- `matched_deal_id` (FK, nullable)
- `matched_project_id` (FK, nullable)
- `match_confidence` (Enum: `high | medium | low | manual | none`)
- `manual_override_at`, `manual_override_user_id` (User hat Match-Zuordnung manuell gesetzt)
- `archived_at` (Soft-Delete im HiveGrid — Provider bleibt unberührt)

Indizes:
- `(tenant_id, mailbox_id, received_at DESC)` — Listen-View
- `(tenant_id, matched_contact_id, received_at DESC)` — Activity-Stream am Kontakt
- `(tenant_id, matched_deal_id, received_at DESC)`
- `(tenant_id, provider_message_id)` unique
- `(tenant_id, provider_thread_id)` für Thread-Aggregation
- Trigram-Index auf `subject`, `from_address`, `body_text` für Volltext-Suche

EmailIntegrationSyncCursor

Pro Mailbox ein Cursor für Initial-Sync und inkrementelle Pulls.

- `id`, `tenant_id`, `mailbox_id` (unique)
- `provider_history_id` (Gmail: `historyId` / Microsoft: `deltaToken`)
- `initial_sync_status` (Enum: `pending | running | completed | failed`)
- `initial_sync_progress` (Integer 0-100, nur während `running`)
- `last_cursor_advance_at`
- `error_text` (nullable)

EmailIntegrationWebhookEvent

Roh-Log der eingegangenen Push-Events vom Provider. Append-only.

- `id`, `tenant_id`, `mailbox_id`, `received_at`
- `provider` (Enum)
- `event_payload` (JSONB — kompletter Provider-Push)

- `processing_status` (Enum: `received | processing | done | failed | discarded`)
- `processed_at`, `error_text`
- `derived_message_ids[]` (welche `EmailIntegrationMessage` aus dem Event entstanden sind)

Retention: 30 Tage, danach gelöscht (außer `failed` → 90 Tage für Diagnose).

Email (HiveGrid-eigene Mails — getrennt von Mailbox-Spiegel)

Repräsentiert *eigene* Mails, die HiveGrid sendet (Booking-Bestätigungen, System-Mails, später Trigger-Mails). Nicht zu verwechseln mit EmailIntegrationMessage (gespiegelte User-Mailbox-Mails).

- `id`, `tenant_id`, `created_at`
- `purpose` (Enum: `booking_confirmation | system_notification | password_reset | invitation | acceptance_protocol | manual_outbound`)
- `from_sender_id` (FK auf TenantSendingDomain bzw. EmailSender — siehe Block C)
- `to_address`, `to_name`
- `cc_addresses[]`, `bcc_addresses[]`
- `subject`, `body_html`, `body_text`
- `delivery_status` (Enum: `queued | sending | delivered | bounced | rejected | failed`)
- `provider_message_id` (von SMTP-Provider)
- `bounce_type` (Enum: `hard | soft | null`)
- `bounce_reason` (Klartext)
- `sent_at`, `delivered_at`, `bounced_at`
- `related_contact_id`, `related_company_id`, `related_deal_id`, `related_project_id` (FKs nullable)
- `archived_at`

Indizes: `(tenant_id, related_contact_id, sent_at DESC)`, `(tenant_id, delivery_status, sent_at DESC)`.

EmailRead

Tracking für *gelesen-durch-Empfänger* (transaktionale Mails von HiveGrid). Pixel-Tracking optional pro Email.

- `id`, `email_id` (FK auf Email), `tenant_id`
- `read_at`
- `reader_ip`, `reader_user_agent` (optional, DSGVO-Hinweis im Footer)

EmailAttachment

Anhänge für sowohl `EmailIntegrationMessage` als auch `Email`.

- `id`, `tenant_id`
- `parent_kind` (Enum: `integration_message | email`)
- `parent_id`
- `filename`, `mime_type`, `size_bytes`
- `storage_path` (Scaleway-Bucket-Pfad: `tenants/{tenantId}/emails/{messageId}/{filename}`)
- `provider_attachment_id` (für Lazy-Download von Provider-Side falls Attachment nicht in Initial-Body geladen)
- `download_status` (Enum: `pending | downloaded | failed`)

TenantSendingDomain

Verifizierte Domain, über die der Tenant *eigene* Mails versendet (transaktional + Marketing).

- `id`, `tenant_id`, `created_at`
- `domain` (z.B. `geerkens.de`)
- `dkim_public_key`, `dkim_private_key_encrypted`, `dkim_selector`
- `dkim_dns_record_text` (was der User in DNS eintragen muss)
- `spf_dns_record_text`

- `dmarc_dns_record_text`
- `verification_status` (Enum: `pending | verifying | verified | failed`)
- `verification_last_check_at`, `verification_error`
- `is_default_for_tenant` (Boolean)
- `archived_at`

Indizes: `(tenant_id, verification_status)`, `(domain)` global-unique.

TenantInboundAddress

System-generierte Inbound-Adresse pro Tenant für Form-Integrationen / Email-zu-Lead.

- `id`, `tenant_id`, `created_at`
- `local_part` (z.B. `tenant-abc-leads`)
- `full_address` (z.B. `tenant-abc-leads@in.hivegrid.de`)
- `purpose` (Enum: `lead_inbox | bcc_logging | reply_routing`)
- `archived_at`

Block B — Mailbox-Connect (OAuth)

Connect-Flow Gmail

1. User in `/{tenant}/settings/integrations/email/` klickt *„Gmail verbinden"*
2. Redirect zu Google-OAuth mit Scopes: `gmail.modify`, `gmail.send`, `gmail.labels`, `userinfo.email`
3. Callback → Tokens werden encrypted gespeichert (`EmailIntegrationMailbox`)
4. System legt Pub/Sub-Subscription auf User-Mailbox an (Gmail-Watch-API)
5. Initial-Sync-Job: letzte 90 Tage Mails laden (oder konfigurierbar — Default 90)
6. `sync_status` → `active`

Connect-Flow Microsoft (Outlook / Microsoft 365)

1. User klickt *„Outlook verbinden"*
2. Redirect zu Microsoft-OAuth mit Scopes: `Mail.ReadWrite`, `Mail.Send`, `User.Read`, `offline_access`
3. Callback → Tokens encrypted gespeichert
4. System legt Microsoft-Graph-Subscription auf `/me/messages` an (3 Tage Lebenszeit → Renewal-Job)
5. Initial-Sync analog Gmail (delta-API, letzte 90 Tage)
6. `sync_status` → `active`

Connect-Flow für Tenant-Sending-Domain (DKIM)

1. Tenant-Admin in `/{tenant}/settings/integrations/email/sending-domain/` klickt *„Domain verbinden"*
2. Eingabe: Domain (z.B. `geerkens.de`)
3. System generiert DKIM-Schlüssel-Paar, baut DNS-TXT-Records (DKIM, SPF, DMARC)
4. UI zeigt **Klartext-Anleitung** mit Copy-to-Clipboard pro Record + Hinweis: *„Diese 3 Einträge bei deinem DNS-Anbieter (z.B. Strato, Hetzner, IONOS) anlegen. Wir prüfen automatisch alle 5 Minuten."*
5. Hintergrund-Job pollt DNS alle 5 Minuten bis Verification-Status `verified` oder Timeout 7 Tage
6. Bei `verified`: Domain ist sendebereit für transaktionale Mails (Modul 9) und Marketing-Mails (Modul 10)

Reconnect / Token-Refresh

- Token-Refresh-Worker läuft alle 30 Min und erneuert Tokens, die in <2h ablaufen
- Bei Refresh-Fehler (User hat OAuth widerrufen) → `sync_status = revoked`, Klartext-Notification an User: *„Deine Gmail-Verbindung wurde unterbrochen — bitte erneut verbinden. [Verbindung erneuern]"*

Disconnect

- User klickt *„Verbindung trennen"*
- Bestätigungs-Dialog mit Klartext: *„Wenn du trennst, werden keine neuen Mails mehr gespiegelt. Bisher gespiegelte Mails bleiben im CRM erhalten. Du kannst die Verbindung später wieder herstellen."*
- System ruft Provider-Token-Revoke-Endpoint auf
- `sync_status = revoked`
- Tokens werden aus DB gelöscht (encrypted, aber Defense-in-Depth)
- Webhook-Subscription beim Provider gekündigt

Block C — Sending-Domain (Pflicht-Fundament für M10)

DKIM-Setup-Anforderungen

- DKIM ist **Pflicht** für jede Sending-Domain (Marketing-Mails ohne DKIM werden von Gmail/Outlook im Spam landen)
- DKIM-Selector pro Tenant generiert (z.B. `hg-{tenant-slug}`)
- Schlüssel-Länge: 2048 Bit
- DKIM-Private-Key wird verschlüsselt gespeichert (KMS-managed-Key in Scaleway)

SPF-Empfehlung

- Tenant erweitert seinen SPF-Record um `include:_spf.hivegrid.de`
- Falls SPF noch nicht existiert → System schlägt vollständigen Record vor

DMARC-Empfehlung

- Empfohlen: `v=DMARC1; p=quarantine; rua=mailto:dmarc-reports@hivegrid.de; ruf=mailto:dmarc-reports@hivegrid.de`
- Optional, nicht Pflicht für M10-Versand

Verification-Logik

- Cron-Job alle 5 Min prüft Domains mit Status `pending` oder `verifying`
- DKIM-Check: DNS-TXT-Lookup auf `{selector}._domainkey.{domain}`
- SPF-Check: DNS-TXT-Lookup auf `{domain}` mit Pattern-Match
- Bei allen 3 OK → Status `verified`
- Bei 7 Tagen ohne Erfolg → Status `failed` + Klartext: *„Wir konnten deine DNS-Einträge nach 7 Tagen nicht finden. [Anleitung erneut anzeigen] [Support kontaktieren]"*

Tenant ohne eigene Domain (Fallback)

- Phase 1: kein Versand möglich ohne verifizierte Sending-Domain für Marketing
- Transaktionale Mails (Booking-Confirmation, etc.) gehen über System-Domain `noreply@hivegrid.de` (mit Tenant-Branding im Klartext)
- Klartext im Marketing-Modul (M10): *„Bevor du Marketing-Mails versendest, musst du eine eigene Sending-Domain einrichten. [Domain einrichten]"*

Block D — Sync-Strategie (Push statt Polling)

Gmail Push (Pub/Sub)

- Pro Mailbox: Watch-API legt Pub/Sub-Subscription auf User-Inbox an
- Pub/Sub liefert HTTPS-Push an `/api/v1/webhooks/email/gmail`
- Push enthält nur `historyId` — kein Mail-Body
- Worker holt mit `historyId` die Diff-Liste über Gmail-API
- Pro neue Message: Body laden, in `EmailIntegrationMessage` einfügen, Auto-Match auslösen

Microsoft Push (Graph-Subscriptions)

- Pro Mailbox: Subscription auf `/me/messages` mit Lifetime 4230 Min (~3 Tage)
- Push an `/api/v1/webhooks/email/microsoft`
- Renewal-Worker erneuert Subscription täglich

Fallback-Polling

- Falls Webhook 4h kein Event liefert (Mailbox aktiv aber kein Push) → Worker pollt 1× pro Stunde als Sicherheitsnetz
- `last_sync_at` wird bei jedem erfolgreichen Sync (Push oder Poll) aktualisiert

Initial-Sync

- Hintergrund-Job lädt letzte 90 Tage Mails
- Progress sichtbar in UI: *„Initial-Sync läuft — 1.234 von 5.678 Mails geladen (22%)"*
- Bei Abbruch resumefähig über `provider_history_id`-Cursor

Klartext-Diagnose im Sync-Status (LP-10)

- `connecting`: *„Wir richten deine Mailbox-Verbindung ein, das dauert wenige Sekunden..."*
- `active`: keine Anzeige (Default-Zustand)
- `paused`: *„Sync pausiert — [Sync starten]"*
- `error`: spezifischer Fehlertext, z.B. *„Wir konnten Google nicht erreichen (HTTP 503). Wir versuchen es erneut in 5 Min. [Manuell prüfen]"*
- `revoked`: *„Deine Mailbox-Verbindung wurde widerrufen. [Erneut verbinden]"*

Block E — Auto-Match-Logik

Mail → Kontakt (über Email-Adresse)

1. Beim Eingang einer Mail: `from_address` wird gegen `ContactEmail` und `Lead.email` (in Tenant-Scope) gematcht
2. **Hard-Match (high confidence):** exakte Email-Übereinstimmung mit Contact → `matched_contact_id` gesetzt + `match_confidence = high`
3. **Hard-Match (high confidence):** Email-Übereinstimmung mit Lead → `matched_contact_id` bleibt null, aber Activity wird am Lead erzeugt
4. **No-Match:** keine Zuordnung → in Inbox sichtbar, manuell zuordbar
5. Bei *outbound* Mails: `to_addresses` werden gematcht (alle Empfänger gegen Contacts geprüft, primärer Match auf erstem Treffer)

Mail → Deal (über Subject-Pattern)

1. Subject wird auf Patterns geprüft:
   - `[Deal #123]` → Deal-ID 123 zuordnen
   - `[Auftrag #456]` → Project-ID 456 zuordnen
   - `[Lead #789]` → Lead-ID 789 zuordnen
2. ID muss im Tenant existieren, sonst Pattern ignoriert
3. Patterns sind Tenant-konfigurierbar in Phase 2 (Phase 1: hartcodiert)
4. Match-Confidence: `high` falls Pattern matched

Mail → Deal (über Kontakt-Verknüpfung — Heuristik)

- Falls `matched_contact_id` gesetzt UND der Kontakt hat genau 1 offenen Deal → Deal automatisch zuordnen mit `match_confidence = medium`
- Falls Kontakt mehrere offene Deals → keine Auto-Zuordnung, manuell in UI

Manuelle Zuordnung

- Im Mail-Detail-View: User kann manuell *„Zu Kontakt zuordnen"* / *„Zu Deal zuordnen"* / *„Zu Projekt zuordnen"*
- `manual_override_at` + `manual_override_user_id` werden gesetzt
- Manuelle Zuordnung überschreibt Auto-Match
- Auto-Match-Logik versucht **nicht erneut** zu matchen, falls `manual_override_at` gesetzt

Edge-Cases

- **Mehrere Contacts mit gleicher Email** (z.B. private + geschäftliche Adresse identisch): primärer Contact (mit `is_primary` Email) wird gewählt; UI-Hinweis: *„Diese Email gehört zu mehreren Kontakten. Aktuell zugeordnet: [Name]. [Anders zuordnen]"*
- **Eigene Mails an eigene Mailbox** (User schreibt sich selbst): `direction = outbound`, kein Auto-Match auf Contact (eigene Email-Adresse ist in `EmailIntegrationMailbox` registriert und wird ausgeschlossen)
- **Newsletter / Bulk-Sender** (List-Unsubscribe-Header vorhanden): keine Auto-Verknüpfung mit Contact, Mail wird in Inbox als *„Newsletter / Bulk"* gekennzeichnet

---

Block F — Inbox-View

Liste (`/{tenant}/emails`)

Zu zeigende Daten pro Eintrag:
- Absender (Name + Email)
- Betreff
- Snippet (erste 100 Zeichen Body)
- Empfangsdatum (relativ, < 7 Tage; absolut sonst)
- Verknüpfter Kontakt / Deal / Projekt (falls gemacht)
- Read/Unread-Indikator
- Anhang-Indikator (falls vorhanden)
- Mailbox-Indikator (falls User mehrere Mailboxen hat)

Benötigte UI-Elemente:
- Tabellarische oder Listen-Anzeige
- Such-Funktion (Volltext über Subject + Body + Absender)
- Filter:
- Mailbox (Multi-Select, wenn User mehrere hat)
- Direction (`inbound | outbound | all`)

- Read-Status (`gelesen | ungelesen | alle`)
- Verknüpft-mit (`Kontakt | Deal | Projekt | nicht verknüpft`)
- Datum (Date-Range-Picker)
- Anhang (Boolean)
- Bulk-Auswahl + Bulk-Actions (Read/Unread markieren, Archivieren)
- Pagination (Cursor-basiert, 50 pro Page)
- Toggle „Archivierte anzeigen"

Operationen:
- Klick auf Zeile → Detail-View
- Bulk-Read-/Unread-Markierung (sync zurück zum Provider)
- Bulk-Archive im HiveGrid (Provider unangetastet)

Performance:
- Index `(tenant_id, mailbox_id, received_at DESC)`
- Trigram-Index für Volltext-Suche
- Permissions-Filter: `email.read_own` zeigt nur eigene Mailboxen, `email.read_all` zeigt alle Tenant-Mailboxen

Detail (`/{tenant}/emails/:id`)

Zu zeigende Daten:
- Vollständige Mail (From, To, CC, Subject, Body-HTML)
- Empfangsdatum + Mailbox
- Anhänge (Liste mit Download-Link)
- Verknüpfter Kontakt / Deal / Projekt (klickbar zu Detail-View)
- Match-Confidence + Hinweis: *„Auto-erkannt"* / *„Manuell zugeordnet von [User]"*
- Thread-Anzeige (alle Mails mit gleichem `provider_thread_id`, chronologisch)

Benötigte UI-Elemente:
- Mail-Body-Render (HTML mit Content-Security-Policy: keine externen Scripts, externe Bilder hinter „Bilder anzeigen"-Toggle)
- Attachment-Liste mit Download
- Thread-Aggregat (collapsed/expanded)
- Reply / Reply-All / Forward-Funktionen (öffnet Compose mit Pre-Fill)
- Verknüpfungs-Editor (Kontakt / Deal / Projekt zuordnen oder ändern)
- Read/Unread-Toggle
- Archivieren-Funktion

Operationen:
- Reply / Reply-All / Forward
- Verknüpfungen ändern (manuelle Override)
- Anhang downloaden
- Mail im Provider öffnen (Deep-Link zu Gmail/Outlook-Web)
- Archivieren

Klartext-Hinweise:
- Externe Bilder geblockt: *„Externe Bilder geladen? [Anzeigen]"*
- Mail nicht zugeordnet: *„Diese Mail ist keinem Kontakt zugeordnet. [Zu Kontakt zuordnen] [Neuen Kontakt anlegen]"*

Block G — Outbound-Compose

Compose-Funktion

Aufruf:
- Aus Kontakt-Detail: *„Email schreiben"* → Empfänger vorausgefüllt mit Primary-Email

- Aus Deal-Detail: *„Email schreiben"* → Empfänger = primärer Kontakt am Deal, Subject pre-fill `[Deal #123]`
- Aus Projekt-Detail: analog mit `[Auftrag #456]`
- Aus Inbox-Detail (Reply/Forward): Empfänger und Subject vorgegeben
- Standalone aus `/{tenant}/emails/new`: leerer Compose

Benötigte UI-Elemente:
- Empfänger-Felder (To, CC, BCC) mit Auto-Complete aus Contacts
- Subject-Feld
- Body-Editor (Rich-Text — Bold/Italic/Link/Liste/Bilder/Inline-Attachments)
- Anhang-Upload (max. 25 MB pro Mail Provider-Limit)
- Mailbox-Auswahl (falls User mehrere Mailboxen hat — Default: Default-Mailbox des Users)
- Send-Funktion + Save-as-Draft-Funktion
- Schedule-Send (Phase 2)

Send-Pfad:
- Mail wird über die verbundene User-Mailbox gesendet (Gmail-API `messages.send` / Microsoft-Graph `sendMail`)
- Provider-Side erscheint die Mail im *Sent*-Folder des Users (nativ)
- HiveGrid spiegelt die gesendete Mail nach erfolgreichem Send als `EmailIntegrationMessage` mit `direction = outbound`
- Activity-Stream-Eintrag wird erzeugt

Klartext-Fehler:
- Mailbox nicht verbunden: *„Du hast keine Mailbox verbunden. [Mailbox verbinden]"*
- Send-Fehler: *„Mail konnte nicht gesendet werden — Google meldet 'Quote exceeded'. In 1h erneut versuchen oder anderen Account verwenden. [Erneut versuchen]"*
- Empfänger ohne Email: *„Der Kontakt hat keine Email-Adresse hinterlegt. [Email ergänzen]"*

BCC-Logging-Adresse:
- Optional: User kann in Settings aktivieren *„Beim Senden BCC an* `tenant-{tenantId}-log@in.hivegrid.de`"
- Für Edge-Case: User schreibt Mail aus nativem Gmail-Client, möchte trotzdem im CRM loggen → BCC-Adresse erzeugt automatisch eine `EmailIntegrationMessage`-Entry mit Auto-Match

---

Block H — Activity-Stream-Integration

Activity-Typen die M9 produziert

- `email_received` — eingehende Mail mit gemachtem Match
- `email_sent` — ausgehende Mail
- `email_unmatched` (Phase 2 — falls UX zeigt: „neue ungeordnete Mail")
- `email_match_changed` — Audit-Event bei manuellem Override

Verknüpfungen

- Activity erscheint im Stream der gematchten Entity (Contact / Deal / Project / Lead)
- `summary_text` Beispiel: *„Eingehende Mail von Max Mustermann: ‚Anfrage Wartung'"*
- `metadata` enthält: Mailbox-ID, Provider-Message-ID, Subject, Snippet
- Klick auf Activity → öffnet Mail-Detail-View

Permission-Wirkung

- Activity sichtbar nur für User mit `email.read_own` (eigene Mailboxen) bzw. `email.read_all`
- Falls User keine Email-Permissions hat: Activity zeigt nur *„Email-Aktivität (nicht einsehbar)"* ohne Details

---

Block I — Permissions

Permission-IDs

Permission	Erlaubt
`email.read_own`	Eigene verbundene Mailbox-Mails sehen
`email.read_all`	Alle Tenant-Mailbox-Mails sehen
`email.write`	Mails verfassen + senden über eigene Mailbox
`email.admin`	Sending-Domain einrichten, Inbound-Adressen verwalten, fremde Mailbox-Connections trennen

Bundle-Mapping

Bundle	Atomare Permissions
`email.none`	(keine)
`email.read_own`	`email.read_own` + `email.write` (eigene Mailbox schreiben können ist Standard)
`email.read_all`	`email.read_own` + `email.read_all` + `email.write`
`email.admin`	alle obigen + `email.admin`

Anwendung in Routes

- `/{tenant}/emails` → `email.read_own` oder `email.read_all`
- `/{tenant}/emails/:id` → analog (mit Owner-Check bei `read_own`: nur Mails aus Mailbox des Users)
- `/{tenant}/emails/new` (Compose) → `email.write` + verbundene Mailbox
- `/{tenant}/settings/integrations/email/` → jeder eingeloggte User für eigene Mailbox; Sending-Domain nur `email.admin` oder `tenant.admin`

Owner-Check bei `read_own`

- User mit `email.read_own` sieht nur Mails aus Mailboxen, deren `owner_user_id = self`
- Auto-Match-Logik bleibt aktiv über alle Mailboxen, aber Anzeige im Activity-Stream eines Kontakts wird permissioned: User mit nur `email.read_own` sieht nur Mails aus seinen Mailboxen am Kontakt

---

Block J — Provider-Adapter (LP-11)

EmailProviderInterface

Gemeinsames Interface für Gmail- und Microsoft-Adapter:

- `connect(authCode) → MailboxConnection`
- `disconnect(mailboxId) → void`
- `fetchMessages(cursor, limit) → MessageBatch`
- `getMessage(providerMessageId) → MessageDetail`
- `sendMessage(MessageDraft) → ProviderMessageId`
- `subscribeToWebhook(mailboxId) → SubscriptionInfo`
- `renewWebhook(subscriptionId) → SubscriptionInfo`
- `revokeWebhook(subscriptionId) → void`

- `markRead(providerMessageId, isRead) → void`
- `downloadAttachment(providerMessageId, attachmentId) → Stream`

Implementierungen

- **GmailAdapter:** spricht Gmail-API + Pub/Sub
- **MicrosoftAdapter:** spricht Microsoft-Graph + Subscription-API

Klartext-Diagnose pro Adapter (LP-10)

Adapter liefern Provider-Fehler übersetzt:
- Gmail `403 quota exceeded` → *„Du hast das tägliche Sende-Limit erreicht. Es wird in [X Stunden] zurückgesetzt."*
- Microsoft `401 token expired` → *„Outlook-Verbindung abgelaufen — bitte erneut verbinden."*
- Generisch unbekannter Fehler → *„Es gab ein Problem mit deiner Mailbox-Verbindung. Wir versuchen es automatisch in 5 Min erneut. [Logs anzeigen]"* (Logs-Anzeige Permission-gegated über `email.admin`)

Block K — Phase 2 / spätere Features

- Erweiterte Filter (gespeicherte Filter, Smart-Folders)
- Bulk-Operations: Multi-Reply, Multi-Forward
- Email-Templates für Outbound (Quick-Reply-Snippets)
- Mail-Merge in Compose (Personalisierung pro Empfänger)
- Schedule-Send (Mail um Datum X versenden)
- Tracking-Pixel für Outbound (Open-Tracking — Phase 2 für transaktionale Mails, in M10 schon Phase 1)
- Snooze-Funktion (Mail kommt in 3 Tagen wieder in Inbox)
- Auto-Forwarding-Regeln (alle Mails von `@geerkens.de` an Sales-Team weiterleiten)
- Custom-Subject-Patterns für Auto-Match (Tenant-konfigurierbar)
- Shared-Inbox (mehrere User auf 1 Mailbox, mit Assignment)

Architektur-Vorschriften (Querschnitt)

- **LP-9:** Requirements-pur, kein Layout
- **LP-10, AV #1:** Klartext-Diagnose bei jedem Sync-Fehler, jeder OAuth-Disconnect, jedem Send-Fehler
- **LP-11:** Provider-Adapter-Pattern, Gmail + Microsoft als erste Implementierungen
- **LP-13:** Mail-Bodies werden in Scaleway-DB gehalten, Attachments in Scaleway-Storage (EU)
- **AV #5 Permissions-aware-Default:** Inbox-Liste filtert automatisch nach `email.read_own` vs. `read_all`
- **AV #6 Audit-Trail:** Manueller Match-Override loggt `audit_event`

Detail-Spec-Status: DRAFT

Erste vollständige Detail-Spec für Modul 9. **Vor Bau:** Validierung der Provider-Webhook-Lifecycle (insbes. Microsoft-3-Tage-Renewal), DKIM-DNS-Check-Implementierung mit echter Domain testen.

Offene Detail-Punkte (nicht-blockierend):
- Konkretes Encryption-KMS-Setup für OAuth-Tokens (Implementierungs-Detail)
- Konkrete Quotas pro Provider (Gmail: 1 Mrd Quota-Units/Tag — reicht weit; Microsoft: 10.000 Mails/Tag pro Mailbox)
- Initial-Sync-Default-Tiefe (90 Tage vorgeschlagen, ggf. Tenant-konfigurierbar)

- Subframe-UI-Designs (separat)
- Verknüpfung mit Modul 24 Globale Suche (Mail-Volltext indexieren)