Date: 12/01/26
For: Eldad Gunshrovitz

Project Brief: AI Email Copilot Chatbot

(Client Demo Project)

Owner	Developer	Version
Aviv Perry	Eldad Gunshrovitz	1.0

Purpose

This is a **demo project that simulates a real client request**. The goal is not to be walked through steps. The goal is that you can **own the solution end to end**: understand the client need, design the automation, implement it in n8n, and deliver it with clean standards, safety, and documentation.

---

1) Client Story (What the client says)

> "I'm drowning in emails. I need AI to manage them, help me sort them, and help me reply faster. I want to chat with something that can look at my inbox and help me decide what to do."

---

2) Technical Interpretation (What we will build)

We will build a **chatbot interface** the client can talk to, where the bot can (with permission) access the client inbox to:

✅ Search emails
✅ Read and summarize emails and threads
✅ Classify and label emails
✅ Draft replies (never send automatically in MVP)
✅ Create drafts inside Gmail
✅ Archive, mark read, move labels
✅ Provide "next actions" and priority

The chatbot uses **AI as the brain** and n8n as the **orchestrator** that safely executes email actions.

---

3) Success Criteria (How we know it works)

User outcomes

- The client can ask questions like:
  - "What is urgent today?"
  - "Summarize emails from John about the contract."
  - "Draft a reply to the latest email in that thread."
  - "Label all newsletters as Newsletter and archive them."
- The bot returns:
  - Clear summaries
  - A recommended action
  - A safe preview of any email action before executing

Quality outcomes

- No repeated actions on the same email (idempotent behavior)
- Failures do not break the workflow (graceful error handling)
- Every action is logged and auditable
- Sensitive data handling is respected (least privilege, redaction option)

---

4) Scope

MVP Features (Must Have)

1. **Chat interface**
   - Telegram is acceptable for MVP (fastest)
   - Alternative: Webhook based chat endpoint if you prefer

2. **Inbox tools**
   - Search emails by query
   - Get latest unread emails summary
   - Read a specific email by id
   - Summarize a thread
   - Apply label, remove label, archive, mark read
   - Create a draft reply in Gmail

3. **Safety layer**
    - Confirmation step before any destructive action (archive, label changes)
    - Confirmation step before creating a draft (show preview)
4. **AI brain contract**
    - Strict JSON outputs for tool execution decisions
    - Parameter validation before calling Gmail actions

Out of Scope for MVP (Not Now)

- Automatically sending emails without confirmation
- Handling attachments deeply (summarize attachment content)
- Long term memory across weeks (beyond basic conversation state)
- Multi inbox accounts in one bot (single account for demo)

---

5) Architecture Expectations (High Level)

Core idea

n8n workflow receives a chat message, AI decides what to do, n8n executes safe Gmail actions, then returns a response.

Recommended components in n8n (You choose exact nodes)

- Trigger: Telegram Trigger or Webhook Trigger
- AI: OpenAI node (chat)
- Email: Gmail nodes (OAuth)
- Logic: Switch, IF, Merge, Set
- Data: Data Store or database for conversation state and idempotency
- Utilities: Code node only when necessary (validation, cleaning, parsing)

---

6) 🔐 Security, Privacy, and Safety Requirements

Authentication and permissions

- Use **OAuth** for Gmail access
- Use **least privilege scopes** required for the MVP
- Credentials must be stored in n8n Credentials, never in code

Data minimization

- Only send to AI what is needed:
  - Subject, From, Date, snippet or cleaned body
  - Avoid full raw HTML whenever possible
- Include a redaction option:
  - Mask phone numbers, id numbers, addresses when detected (basic patterns are enough)

No silent destructive actions

- Must require confirmation for:
  - Archive
  - Mark as read
  - Apply or remove labels in bulk
  - Create draft
- Provide a “dry run” preview for bulk operations:
  - Show count and a few examples

Audit trail

- Log every action:
  - timestamp
  - chat user
  - intended action
  - parameters
  - success or failure
- Logs should be easy to inspect later (Data Store, sheet, database, or n8n execution logs plus structured message)

---

7) AI Behavior Contract (Very important)

System behavior rules

The AI must:

- Ask clarifying questions when user request is ambiguous
- Prefer safe actions: summarize first, then act
- Never fabricate email content
- Never claim it executed an email action unless n8n actually confirmed execution

Output format

For any tool execution decision, the AI must return **JSON only** using a schema like this:

```
{
  "intent": "search | summarize | draft_reply | label | archive |
mark_read | ask_clarifying",
  "confidence": 0.0,
  "needs_confirmation": true,
  "message_to_user": "string",

  "email_insights": {
    "sentiment": "positive | negative | angry | interested | neutral |
confused",
    "urgency": 1,
    "summary": "string",
    "action_items": {
      "exists": false,
      "items": []
    }
  },

  "action": {
    "type": "gmail.search | gmail.get | gmail.thread | gmail.draft |
gmail.label | gmail.archive | gmail.mark_read",
    "params": {}
  }
}
```

Notes

- needs_confirmation must be true for anything destructive or bulk
- confidence is used for safety: if low, ask clarifying

Tool parameter validation

Before executing action.params, n8n must validate:

- required fields exist
- types are correct
- query is not empty
- bulk actions include a limit and preview first

8) Gmail Operations Requirements

Search

- Support Gmail search query syntax in a simple way
- Provide safe defaults:
  - limit results, for example 5 or 10
  - sort by newest

Summarize

- Summarize individual email and thread
- Strip quoted history when possible
- Provide:
  - 1 line summary
  - urgency score 1 to 5
  - category
  - recommended next action

Draft reply

- Draft must include:
  - greeting
  - short answer
  - next step question or call to action
  - signature placeholder
- Draft must be shown to user for approval and optionally edited through chat

---

9) Reliability and Error Handling

Idempotency

Prevent duplicates:

- Track processed email ids per action
- If the same request repeats due to retry, do not re execute blindly

Rate limits and retries

- Handle transient failures:
  - exponential backoff on 429 and 5xx where appropriate
- If retry fails:
  - return a helpful user message
  - log the error with context

Graceful fallbacks

- If AI fails or returns invalid JSON:
  - do not execute actions
  - ask the user to rephrase
  - notify with error details in logs

---

10) n8n Engineering Standards (House rules)

Node naming convention

Every node must follow:
[Service] [Action] [Entity]

Examples:

- Gmail Search Emails
- OpenAI Decide Intent
- IF Needs Confirmation
- Telegram Send Response
- DataStore Save Session

Workflow layout

- Group by sections with sticky notes:
  - Input
  - Intent decision
  - Validation and safety
  - Execution
  - Response
  - Errors and logging

Code node policy

- Only for:
  - JSON parsing and validation
  - cleaning email content
  - redaction
- Keep code short, readable, and commented

---

11) Deliverables (What you must submit)

1. **Working n8n workflow**
   - Runs end to end
   - Handles both happy path and failures
2. **Exported workflow JSON**
3. **System prompt**
   - The exact system instructions used for the AI brain
4. **Test evidence**
   - Screenshots or logs of at least 10 test scenarios
5. **Short documentation**
   - Setup steps
   - Credentials required
   - How confirmation works
   - Known limitations

---

12) Test Plan (Minimum scenarios)

You must test and document at least these:

1. “Summarize my latest unread emails”
2. “Find emails from X about invoice”
3. “Summarize this thread” (multi message)
4. “Draft a reply saying yes and propose 2 times”
5. “Label newsletters and archive them” with confirmation
6. AI returns invalid JSON (simulate) and system recovers safely

7. Gmail returns rate limit and workflow retries safely
8. Email body is HTML heavy and still produces usable summary
9. Very short email, one line, no crash
10. Attempt a destructive action without confirmation, must be blocked

---

13) 🧑‍💻 Working Agreement (How we avoid friction)

Autonomy

- You own the design and implementation decisions.
- You present the final approach and why.

Communication

- One daily async update in our Slack chat:
  - What you did
  - What is blocked
  - Answeres for questions you must get from me (including urgency level)
  - What you plan next

Escalation rule

- If you are stuck more than 60 minutes, write a short message:
  - what you tried
  - what failed
  - your current hypothesis
    Then we do a quick unblock.

---

14) Effort Estimate (Internal)

- You have to tell me your estimation of time for this project before you start it.
- Then I will fill in the following:
  - MVP target effort: **__ to __ hours** depending on polish and testing quality.

---

15) Definition of Done (Final checklist) ✅

Done means:

- Chat can search, summarize, label, archive, mark read, draft reply
- Confirmation gate exists and works
- Strict JSON contract is enforced
- Idempotency prevents duplicates
- Errors are handled and logged
- Documentation and workflow export are delivered

📌 Final note

This project is designed to evaluate how you build real client grade automations: safety, clarity, maintainability, and communication, not only “it runs on my machine”.